BILL AND NATHAN, RECORD LECTURE!!!!

BILL RECORD LECTURE!!!

Upper and Lower Bounds (PCP) on Approx For MAX3SAT

We Assume $\cdots$

In this section we assume $\mathrm{P} \neq \mathrm{NP}$.

We Assume · · ·

In this section we assume P $\neq$ NP.

If we say

The Hokey Pokey cannot be approx better than BLAH

We Assume $\cdots$

In this section we assume P $\neq$ NP.

If we say

The Hokey Pokey cannot be approx better than BLAH

We mean

If can approx The Hokey Pokey better than BLAH then $\mathrm{P} = \mathrm{NP}$

We Assume $\cdots$

In this section we assume P $\neq$ NP.

If we say

The Hokey Pokey cannot be approx better than BLAH

We mean

If can approx The Hokey Pokey better than BLAH then $\mathrm{P} = \mathrm{NP}$

If we say **Alg A** we mean **Poly time Alg A**.

We Assume $\cdots$

In this section we assume P $\neq$ NP.

If we say

The Hokey Pokey cannot be approx better than BLAH

We mean

If can approx The Hokey Pokey better than BLAH then $\mathrm{P} = \mathrm{NP}$

If we say **Alg A** we mean **Poly time Alg A**.

If we say **rand Alg A** we mean **Randomized Poly time Alg A**.

MAX3SAT

MAX3SAT

MAX3SAT

MAX3SAT

1. **Input** $\phi = C_1 \wedge \cdots \wedge C_m$, each C_i is a $\vee$ of 3 literals.

MAX3SAT

MAX3SAT

1. **Input** $\phi = C_1 \wedge \cdots \wedge C_m$, each C_i is a $\vee$ of 3 literals.
2. **Output** The max number of clauses that can be satisfied.

MAX3SAT

MAX3SAT

1. **Input** $\phi = C_1 \wedge \cdots \wedge C_m$, each C_i is a $\vee$ of 3 literals.
2. **Output** The max number of clauses that can be satisfied.

Is there a $\delta < 1$ and an alg A such that

MAX3SAT

MAX3SAT

1. **Input** $\phi = C_1 \wedge \cdots \wedge C_m$, each C_i is a $\vee$ of 3 literals.
2. **Output** The max number of clauses that can be satisfied.

Is there a $\delta < 1$ and an alg A such that

$$A(\phi) \geq (1-\delta)\mathrm{MAX3SAT}(\phi)$$

MAX3SAT

MAX3SAT

1. **Input** $\phi = C_1 \wedge \cdots \wedge C_m$, each C_i is a $\vee$ of 3 literals.
2. **Output** The max number of clauses that can be satisfied.

Is there a $\delta < 1$ and an alg A such that

$$A(\phi) \geq (1-\delta)\mathrm{MAX3SAT}(\phi)$$

Yes.

MAX3SAT

$\mathrm{MAX3SAT}$

1. **Input** $\phi = C_1 \wedge \cdots \wedge C_m$, each C_i is a $\vee$ of 3 literals.
2. **Output** The max number of clauses that can be satisfied.

Is there a $\delta < 1$ and an alg A such that

$$A(\phi) \geq (1-\delta)\mathrm{MAX3SAT}(\phi)$$

Yes.

Next Slide

Approx for MAX3SAT

Thm $(\exists)$ **rand** alg A st $A(\phi) \geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

Approx for MAX3SAT

Thm $(\exists)$ **rand** alg A st $A(\phi) \geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

1. Input $\phi = C_1 \wedge \cdots \wedge C_m$.

Approx for MAX3SAT

Thm $(\exists)$ **rand** alg A st $A(\phi) \geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

1. Input $\phi = C_1 \wedge \cdots \wedge C_m$.
2. Assign each var T or F at Random.

Approx for MAX3SAT

Thm ($\exists$) **rand** alg A st $A(\phi) \ge \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

1. Input $\phi = C_1 \wedge \cdots \wedge C_m$.
2. Assign each var T or F at Random.

Its just that easy!

Approx for MAX3SAT

Thm ($\exists$) **rand** alg A st $A(\phi) \geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

1. Input $\phi = C_1 \wedge \cdots \wedge C_m$.
2. Assign each var T or F at Random.

Its just that easy! Why does this work?

Approx for MAX3SAT

Thm $(\exists)$ **rand** alg A st $A(\phi) \geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

1. Input $\phi = C_1 \wedge \cdots \wedge C_m$.
2. Assign each var T or F at Random.

Its just that easy! Why does this work?

Let C be a clause. The prob that C is satisfied is $\frac{7}{8}$.

Approx for MAX3SAT

Thm $(\exists)$ **rand** alg A st $A(\phi) \geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

1. Input $\phi = C_1 \wedge \cdots \wedge C_m$.
2. Assign each var T or F at Random.

Its just that easy! Why does this work?

Let C be a clause. The prob that C is satisfied is $\frac{7}{8}$.
By Lin of ExpV, expected number of C_i satisfied is $\frac{7m}{8}$.

Approx for MAX3SAT

Thm $(\exists)$ **rand** alg A st $A(\phi) \geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

1. Input $\phi = C_1 \wedge \cdots \wedge C_m$.
2. Assign each var T or F at Random.

Its just that easy! Why does this work?

Let C be a clause. The prob that C is satisfied is $\frac{7}{8}$.
By Lin of ExpV, expected number of C_i satisfied is $\frac{7m}{8}$.
Note that $\mathrm{MAX3SAT} \leq m$.

Approx for MAX3SAT

Thm ($\exists$) **rand** alg A st $A(\phi) \ge \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

1. Input $\phi = C_1 \wedge \cdots \wedge C_m$.
2. Assign each var T or F at Random.

Its just that easy! Why does this work?

Let C be a clause. The prob that C is satisfied is $\frac{7}{8}$.
By Lin of ExpV, expected number of C_i satisfied is $\frac{7m}{8}$.
Note that $\mathrm{MAX3SAT} \le m$.
Hence $A(\phi) \ge \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

Approx for MAX3SAT

Thm $(\exists)$ **rand** alg A st $A(\phi) \geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

1. Input $\phi = C_1 \wedge \cdots \wedge C_m$.
2. Assign each var T or F at Random.

Its just that easy! Why does this work?

Let C be a clause. The prob that C is satisfied is $\frac{7}{8}$.
By Lin of ExpV, expected number of C_i satisfied is $\frac{7m}{8}$.
Note that $\mathrm{MAX3SAT} \leq m$.
Hence $A(\phi) \geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

Note This rand alg can be made det by method of cond prob.

Approx for Variants of MAX3SAT

Approx for Variants of MAX3SAT

1. If $(\forall i)[|C_i| = 3]$ then have **easy** rand alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

Approx for Variants of MAX3SAT

1. If $(\forall i)[|C_i| = 3]$ then have **easy** rand alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.
2. If $(\forall i)[|C_i| = 3]$ then have **medium** det alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

Approx for Variants of MAX3SAT

1. If $(\forall i)[|C_i| = 3]$ then have **easy** rand alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.
2. If $(\forall i)[|C_i| = 3]$ then have **medium** det alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.
3. If $(\forall i)[|C_i| \leq 3]$ then have **easy** rand alg returns $\geq \frac{1}{2}\mathrm{MAX3SAT}(\phi)$.

Approx for Variants of MAX3SAT

1. If $(\forall i)[|C_i| = 3]$ then have **easy** rand alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.
2. If $(\forall i)[|C_i| = 3]$ then have **medium** det alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.
3. If $(\forall i)[|C_i| \leq 3]$ then have **easy** rand alg returns $\geq \frac{1}{2}\mathrm{MAX3SAT}(\phi)$.
4. If $(\forall i)[|C_i| \leq 3]$ then have **medium** det alg returns $\geq \frac{1}{2}\mathrm{MAX3SAT}(\phi)$.

Approx for Variants of MAX3SAT

1. If $(\forall i)[|C_i| = 3]$ then have **easy** rand alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.
2. If $(\forall i)[|C_i| = 3]$ then have **medium** det alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.
3. If $(\forall i)[|C_i| \leq 3]$ then have **easy** rand alg returns $\geq \frac{1}{2}\mathrm{MAX3SAT}(\phi)$.
4. If $(\forall i)[|C_i| \leq 3]$ then have **medium** det alg returns $\geq \frac{1}{2}\mathrm{MAX3SAT}(\phi)$.
5. If $(\forall i)[|C_i| \leq 3]$ then have **hard** rand alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

Approx for Variants of MAX3SAT

1. If $(\forall i)[|C_i| = 3]$ then have **easy** rand alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.
2. If $(\forall i)[|C_i| = 3]$ then have **medium** det alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.
3. If $(\forall i)[|C_i| \leq 3]$ then have **easy** rand alg returns $\geq \frac{1}{2}\mathrm{MAX3SAT}(\phi)$.
4. If $(\forall i)[|C_i| \leq 3]$ then have **medium** det alg returns $\geq \frac{1}{2}\mathrm{MAX3SAT}(\phi)$.
5. If $(\forall i)[|C_i| \leq 3]$ then have **hard** rand alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

People tried to get an app-alg to return $\geq (\frac{7}{8} + \epsilon)\mathrm{MAX3SAT}(\phi)$.

Approx for Variants of MAX3SAT

1. If $(\forall i)[|C_i| = 3]$ then have **easy** rand alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.
2. If $(\forall i)[|C_i| = 3]$ then have **medium** det alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.
3. If $(\forall i)[|C_i| \leq 3]$ then have **easy** rand alg returns $\geq \frac{1}{2}\mathrm{MAX3SAT}(\phi)$.
4. If $(\forall i)[|C_i| \leq 3]$ then have **medium** det alg returns $\geq \frac{1}{2}\mathrm{MAX3SAT}(\phi)$.
5. If $(\forall i)[|C_i| \leq 3]$ then have **hard** rand alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

People tried to get an app-alg to return $\geq (\frac{7}{8} + \epsilon)\mathrm{MAX3SAT}(\phi)$.
Did they succeed?

Approx for Variants of MAX3SAT

1. If $(\forall i)[|C_i| = 3]$ then have **easy** rand alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.
2. If $(\forall i)[|C_i| = 3]$ then have **medium** det alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.
3. If $(\forall i)[|C_i| \leq 3]$ then have **easy** rand alg returns $\geq \frac{1}{2}\mathrm{MAX3SAT}(\phi)$.
4. If $(\forall i)[|C_i| \leq 3]$ then have **medium** det alg returns $\geq \frac{1}{2}\mathrm{MAX3SAT}(\phi)$.
5. If $(\forall i)[|C_i| \leq 3]$ then have **hard** rand alg returns $\geq \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

People tried to get an app-alg to return $\geq (\frac{7}{8} + \epsilon)\mathrm{MAX3SAT}(\phi)$.
Did they succeed? No.

Approx for Variants of MAX3SAT

1. If $(\forall i)[|C_i| = 3]$ then have **easy** rand alg returns $\ge \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.
2. If $(\forall i)[|C_i| = 3]$ then have **medium** det alg returns $\ge \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.
3. If $(\forall i)[|C_i| \le 3]$ then have **easy** rand alg returns $\ge \frac{1}{2}\mathrm{MAX3SAT}(\phi)$.
4. If $(\forall i)[|C_i| \le 3]$ then have **medium** det alg returns $\ge \frac{1}{2}\mathrm{MAX3SAT}(\phi)$.
5. If $(\forall i)[|C_i| \le 3]$ then have **hard** rand alg returns $\ge \frac{7}{8}\mathrm{MAX3SAT}(\phi)$.

People tried to get an app-alg to return $\ge (\frac{7}{8} + \epsilon)\mathrm{MAX3SAT}(\phi)$.
Did they succeed? No. Now What?

There is a Limit To How Well You Can Approx

We will show there is **some** $\boldsymbol{\delta < \frac{1}{8}}$ such that there is NO app-alg that returns

$$\geq (1-\delta)\mathrm{MAX3SAT}(\phi).$$

There is a Limit To How Well You Can Approx

We will show there is **some** $\boldsymbol{\delta < \frac{1}{8}}$ such that there is NO app-alg that returns

$$\geq (1-\delta)\mathrm{MAX3SAT}(\phi).$$

Hence cannot keep getting better and better approx.

There is a Limit To How Well You Can Approx

We will show there is **some $\boldsymbol{\delta < \frac{1}{8}}$** such that there is NO app-alg that returns

$$\geq (1-\delta)\mathrm{MAX3SAT}(\phi).$$

Hence cannot keep getting better and better approx.

Consequence $\exists \epsilon < \frac{1}{8}$, $\neg\exists$ alg A, $A(\phi) \geq (\frac{7}{8}+\epsilon)\mathrm{MAX3SAT}(\phi)$.

There is a Limit To How Well You Can Approx

We will show there is **some** $\boldsymbol{\delta < \frac{1}{8}}$ such that there is NO app-alg that returns

$$\geq (1-\delta)\mathrm{MAX3SAT}(\phi).$$

Hence cannot keep getting better and better approx.

Consequence $\exists \epsilon < \frac{1}{8}$, $\neg\exists$ alg A, $A(\phi) \geq (\frac{7}{8}+\epsilon)\mathrm{MAX3SAT}(\phi)$.

The value of ϵ is buried in the machinery of PCP though it could be determined.

There is a Limit To How Well You Can Approx

We will show there is **some $\delta < \frac{1}{8}$** such that there is NO app-alg that returns

$$\geq (1-\delta)\mathrm{MAX3SAT}(\phi).$$

Hence cannot keep getting better and better approx.

Consequence $\exists \epsilon < \frac{1}{8}$, $\neg\exists$ alg A, $A(\phi) \geq (\frac{7}{8}+\epsilon)\mathrm{MAX3SAT}(\phi)$.

The value of ϵ is buried in the machinery of PCP though it could be determined.

Likely end up with something like:

There is a Limit To How Well You Can Approx

We will show there is **some $\delta < \frac{1}{8}$** such that there is NO app-alg that returns

$$\geq (1-\delta)\mathrm{MAX3SAT}(\phi).$$

Hence cannot keep getting better and better approx.

Consequence $\exists \epsilon < \frac{1}{8}$, $\neg\exists$ alg A, $A(\phi) \geq (\frac{7}{8}+\epsilon)\mathrm{MAX3SAT}(\phi)$.

The value of ϵ is buried in the machinery of PCP though it could be determined.

Likely end up with something like:
There is **no Alg** A such that

There is a Limit To How Well You Can Approx

We will show there is **some** $\boldsymbol{\delta < \frac{1}{8}}$ such that there is NO app-alg that returns

$$\geq (1 - \delta)\mathrm{MAX3SAT}(\phi).$$

Hence cannot keep getting better and better approx.

Consequence $\exists \epsilon < \frac{1}{8}$, $\neg\exists$ alg A, $A(\phi) \geq (\frac{7}{8} + \epsilon)\mathrm{MAX3SAT}(\phi)$.

The value of ϵ is buried in the machinery of PCP though it could be determined.

Likely end up with something like:
There is **no Alg** A such that

$$A(\phi) \geq \frac{10^{40} - 1}{10^{40}}\mathrm{MAX3SAT}(\phi).$$

There is a Limit To How Well You Can Approx

We will show there is **some $\delta < \frac{1}{8}$** such that there is NO app-alg that returns

$$\geq (1-\delta)\mathrm{MAX3SAT}(\phi).$$

Hence cannot keep getting better and better approx.

Consequence $\exists \epsilon < \frac{1}{8}$, $\neg\exists$ alg A, $A(\phi) \geq (\frac{7}{8}+\epsilon)\mathrm{MAX3SAT}(\phi)$.

The value of ϵ is buried in the machinery of PCP though it could be determined.

Likely end up with something like:
There is **no Alg** A such that

$$A(\phi) \geq \frac{10^{40}-1}{10^{40}}\mathrm{MAX3SAT}(\phi).$$

(An alg that does better and better is a **Poly Time Approx Scheme(PTAS)**. We show there is no PTAS for MAX3SAT.)

Better Lower Bounds Are Known

Thm
$\forall \epsilon > 0$, $\neg\exists$ alg A, $A(\phi) \geq (\frac{7}{8} + \epsilon)\mathrm{MAX3SAT}(\phi)$.

Better Lower Bounds Are Known

Thm

$\forall \epsilon > 0$, $\neg\exists$ alg A, $A(\phi) \geq (\frac{7}{8} + \epsilon)\mathrm{MAX3SAT}(\phi)$.

So can't even do a wee bit better,

Better Lower Bounds Are Known

Thm
$\forall \epsilon > 0$, $\neg\exists$ alg A, $A(\phi) \geq (\frac{7}{8} + \epsilon)\mathrm{MAX3SAT}(\phi)$.
So can't even do a wee bit better,

If Erika says she has an alg that returns $\geq (\frac{7}{8} + \frac{1}{10^{40}})\mathrm{MAX3SAT}(\phi)$
then either

Better Lower Bounds Are Known

Thm
$\forall\epsilon > 0$, $\neg\exists$ alg A, $A(\phi) \ge (\frac{7}{8} + \epsilon)\mathrm{MAX3SAT}(\phi)$.
So can't even do a wee bit better,

If Erika says she has an alg that returns $\ge (\frac{7}{8} + \frac{1}{10^{40}})\mathrm{MAX3SAT}(\phi)$ then either (a) Erika has proven $\mathrm{P} = \mathrm{NP}$ or

Better Lower Bounds Are Known

Thm
$\forall \epsilon > 0$, $\neg\exists$ alg A, $A(\phi) \geq (\frac{7}{8} + \epsilon)\mathrm{MAX3SAT}(\phi)$.
So can't even do a wee bit better,

If Erika says she has an alg that returns $\geq (\frac{7}{8} + \frac{1}{10^{40}})\mathrm{MAX3SAT}(\phi)$ then either (a) Erika has proven $\mathrm{P} = \mathrm{NP}$ or (b) Erika is mistaken.

Better Lower Bounds Are Known

Thm
$\forall\epsilon > 0$, $\neg\exists$ alg A, $A(\phi) \ge (\frac{7}{8} + \epsilon)\mathrm{MAX3SAT}(\phi)$.
So can't even do a wee bit better,

If Erika says she has an alg that returns $\ge (\frac{7}{8} + \frac{1}{10^{40}})\mathrm{MAX3SAT}(\phi)$ then either (a) Erika has proven $\mathrm{P} = \mathrm{NP}$ or (b) Erika is mistaken.

Yet another example of the explanatory power of $\mathrm{P} \ne \mathrm{NP}$

Better Lower Bounds Are Known

Thm
$\forall \epsilon > 0$, $\neg\exists$ alg A, $A(\phi) \geq (\frac{7}{8} + \epsilon)\mathrm{MAX3SAT}(\phi)$.
So can't even do a wee bit better,

If Erika says she has an alg that returns $\geq (\frac{7}{8} + \frac{1}{10^{40}})\mathrm{MAX3SAT}(\phi)$ then either (a) Erika has proven $\mathrm{P} = \mathrm{NP}$ or (b) Erika is mistaken.

Yet another example of the explanatory power of $\mathrm{P} \neq \mathrm{NP}$

Note that

Better Lower Bounds Are Known

Thm
$\forall \epsilon > 0$, $\neg\exists$ alg A, $A(\phi) \geq (\frac{7}{8} + \epsilon)\mathrm{MAX3SAT}(\phi)$.
So can't even do a wee bit better,

If Erika says she has an alg that returns $\geq (\frac{7}{8} + \frac{1}{10^{40}})\mathrm{MAX3SAT}(\phi)$ then either (a) Erika has proven $\mathrm{P} = \mathrm{NP}$ or (b) Erika is mistaken.

Yet another example of the explanatory power of $\mathrm{P} \neq \mathrm{NP}$

Note that

1. The rand and poly app-algs that got $\frac{7}{8}\mathrm{MAX3SAT}(\phi)$ are easy.

Better Lower Bounds Are Known

Thm
$\forall \epsilon > 0$, $\neg\exists$ alg A, $A(\phi) \ge (\frac{7}{8} + \epsilon)\mathrm{MAX3SAT}(\phi)$.
So can't even do a wee bit better,

If Erika says she has an alg that returns $\ge (\frac{7}{8} + \frac{1}{10^{40}})\mathrm{MAX3SAT}(\phi)$ then either (a) Erika has proven $\mathrm{P} = \mathrm{NP}$ or (b) Erika is mistaken.

Yet another example of the explanatory power of $\mathrm{P} \ne \mathrm{NP}$

Note that

1. The rand and poly app-algs that got $\frac{7}{8}\mathrm{MAX3SAT}(\phi)$ are easy.
2. The lower bound uses PCP machinery.

Better Lower Bounds Are Known

Thm
$\forall \epsilon > 0$, $\neg\exists$ alg A, $A(\phi) \geq (\frac{7}{8} + \epsilon)\mathrm{MAX3SAT}(\phi)$.
So can't even do a wee bit better,

If Erika says she has an alg that returns $\geq (\frac{7}{8} + \frac{1}{10^{40}})\mathrm{MAX3SAT}(\phi)$ then either (a) Erika has proven $\mathrm{P} = \mathrm{NP}$ or (b) Erika is mistaken.

Yet another example of the explanatory power of $\mathrm{P} \neq \mathrm{NP}$

Note that

1. The rand and poly app-algs that got $\frac{7}{8}\mathrm{MAX3SAT}(\phi)$ are easy.
2. The lower bound uses PCP machinery.
3. The alg and the lower bounds have **nothing to do with each other** and yet yield **matching** upper and lower bounds at $\frac{7}{8}$.

Turning PCP's into Formulas

Recall Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$ such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.

Turning PCP's into Formulas

Recall Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$ such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $x \in \{0, 1\}^n$. This is the input to the PCP.

Turning PCP's into Formulas

Recall Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$ such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $x \in \{0, 1\}^n$. This is the input to the PCP.
We form a Boolean formula as follows.

Turning PCP's into Formulas

Recall Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$ such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $x \in \{0,1\}^n$. This is the input to the PCP.
We form a Boolean formula as follows.
The Vars For every $\tau\sigma \in \{0,1\}^{d \lg n + q}$ one can run the PCP with random string τ and bit-answers σ. From these simulations you can find **all** possible bit-queries. There are $\leq 2^{d \lg n + q} = 2^q n^d$ bit queries. These will be variables.

Turning PCP's into Formulas

Recall Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$ such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $x \in \{0,1\}^n$. This is the input to the PCP.
We form a Boolean formula as follows.
The Vars For every $\tau\sigma \in \{0,1\}^{d \lg n + q}$ one can run the PCP with random string τ and bit-answers σ. From these simulations you can find **all** possible bit-queries. There are $\le 2^{d \lg n + q} = 2^q n^d$ bit queries. These will be variables.

Parts of the Formula For every $\tau \in \{0,1\}^{d \lg n}$ we form ψ_τ.

Turning PCP's into Formulas

Recall Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$ such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $x \in \{0,1\}^n$. This is the input to the PCP.
We form a Boolean formula as follows.
The Vars For every $\tau\sigma \in \{0,1\}^{d \lg n + q}$ one can run the PCP with random string τ and bit-answers σ. From these simulations you can find **all** possible bit-queries. There are $\le 2^{d \lg n + q} = 2^q n^d$ bit queries. These will be variables.

Parts of the Formula For every $\tau \in \{0,1\}^{d \lg n}$ we form ψ_τ.
Use τ as the random string. Simulate all possible query paths to find the relevant vars.

Turning PCP's into Formulas

Recall Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$ such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $x \in \{0,1\}^n$. This is the input to the PCP.
We form a Boolean formula as follows.
The Vars For every $\tau\sigma \in \{0,1\}^{d \lg n + q}$ one can run the PCP with random string τ and bit-answers σ. From these simulations you can find **all** possible bit-queries. There are $\le 2^{d \lg n + q} = 2^q n^d$ bit queries. These will be variables.

Parts of the Formula For every $\tau \in \{0,1\}^{d \lg n}$ we form ψ_τ.
Use τ as the random string. Simulate all possible query paths to find the relevant vars.
ψ_τ is the formula on those vars that is TRUE exactly when that setting of the variables makes this path accept.

A Very Small Example

Imagine the following.

A Very Small Example

Imagine the following.
Using $\tau = 1101$ the PCP will query bit 17.

A Very Small Example

Imagine the following.
Using $\tau = 1101$ the PCP will query bit 17.
If bit 17 is 1 then query bit 84. If bit 17 is 0 then query bit 5.

A Very Small Example

Imagine the following.
Using $\tau = 1101$ the PCP will query bit 17.
If bit 17 is 1 then query bit 84. If bit 17 is 0 then query bit 5.
If bit 17 is 1 and then bit 84 is 1 then accept.
If bit 17 is 0 and then bit 5 is 0 then accept.
All else reject.

A Very Small Example

Imagine the following.
Using $\tau = 1101$ the PCP will query bit 17.
If bit 17 is 1 then query bit 84. If bit 17 is 0 then query bit 5.
If bit 17 is 1 and then bit 84 is 1 then accept.
If bit 17 is 0 and then bit 5 is 0 then accept.
All else reject.
$\psi_{1101} = (q_{17} \wedge q_{84}) \vee (\neg q_{17} \wedge \neg q_5)$.

Max Number of Clauses

In general case we will turn ψ_τ into a $3\mathrm{CNF}$.

Max Number of Clauses

In general case we will turn ψ_τ into a $3\mathrm{CNF}$.

We do not have any control over how many clauses ψ_τ will have.

Max Number of Clauses

In general case we will turn ψ_τ into a $3\mathrm{CNF}$.

We do not have any control over how many clauses ψ_τ will have.

But we do know that it uses $\leq 2^q$ variables.

Max Number of Clauses

In general case we will turn ψ_τ into a $3\mathrm{CNF}$.

We do not have any control over how many clauses ψ_τ will have.

But we do know that it uses $\leq 2^q$ variables.

Def $C(q)$ is max numb of clauses a $3\mathrm{CNF}$ fml on 2^q vars has.

Max Number of Clauses

In general case we will turn ψ_τ into a $3\mathrm{CNF}$.

We do not have any control over how many clauses ψ_τ will have.

But we do know that it uses $\leq 2^q$ variables.

Def $C(q)$ is max numb of clauses a $3\mathrm{CNF}$ fml on 2^q vars has.

Note Since q is a constant, $C(q)$ is a constant.

Max Number of Clauses

In general case we will turn ψ_τ into a $3\mathrm{CNF}$.

We do not have any control over how many clauses ψ_τ will have.

But we do know that it uses $\leq 2^q$ variables.

Def $C(q)$ is max numb of clauses a $3\mathrm{CNF}$ fml on 2^q vars has.

Note Since q is a constant, $C(q)$ is a constant.

We will use $C(q)$ later.

Final Formula

Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$
Let $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.

Final Formula

Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$
Let $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $x \in \{0,1\}^n$. This is the input to the PCP.

Final Formula

Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$
Let $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $x \in \{0,1\}^n$. This is the input to the PCP.
We have said how to take $\tau \in \{0,1\}^{d \lg n}$ and form ψ_τ.

Final Formula

Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$
Let $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $x \in \{0,1\}^n$. This is the input to the PCP.
We have said how to take $\tau \in \{0,1\}^{d \lg n}$ and form ψ_τ.

1. ψ_τ is on $\leq 2^q$ vars, a constant. Rewrite ψ_τ as a $3\mathrm{CNF}$.

Final Formula

Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$
Let $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $x \in \{0,1\}^n$. This is the input to the PCP.
We have said how to take $\tau \in \{0,1\}^{d \lg n}$ and form ψ_τ.

1. ψ_τ is on $\leq 2^q$ vars, a constant. Rewrite ψ_τ as a $3\mathrm{CNF}$.
2. ψ_τ has $\leq C(q)$ clauses. Add clauses of the form $(x \vee x \vee x)$ with new vars x to get exactly $C(q)$ clauses.

Final Formula

Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$
Let $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $x \in \{0,1\}^n$. This is the input to the PCP.
We have said how to take $\tau \in \{0,1\}^{d \lg n}$ and form ψ_τ.

1. ψ_τ is on $\le 2^q$ vars, a constant. Rewrite ψ_τ as a $3\mathrm{CNF}$.
2. ψ_τ has $\le C(q)$ clauses. Add clauses of the form $(x \vee x \vee x)$ with new vars x to get exactly $C(q)$ clauses.
3. Let ψ_x be the $\bigwedge$ of all the ψ_τ.

Final Formula

Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$
Let $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $x \in \{0,1\}^n$. This is the input to the PCP.
We have said how to take $\tau \in \{0,1\}^{d \lg n}$ and form ψ_τ.

1. ψ_τ is on $\leq 2^q$ vars, a constant. Rewrite ψ_τ as a $3\mathrm{CNF}$.
2. ψ_τ has $\leq C(q)$ clauses. Add clauses of the form $(x \vee x \vee x)$ with new vars x to get exactly $C(q)$ clauses.
3. Let ψ_x be the $\bigwedge$ of all the ψ_τ.
4. Note that ψ_x is $3\mathrm{CNF}$.

Final Formula

Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$
Let $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $x \in \{0,1\}^n$. This is the input to the PCP.
We have said how to take $\tau \in \{0,1\}^{d \lg n}$ and form ψ_τ.

1. ψ_τ is on $\le 2^q$ vars, a constant. Rewrite ψ_τ as a $3\mathrm{CNF}$.
2. ψ_τ has $\le C(q)$ clauses. Add clauses of the form $(x \vee x \vee x)$ with new vars x to get exactly $C(q)$ clauses.
3. Let ψ_x be the $\bigwedge$ of all the ψ_τ.
4. Note that ψ_x is $3\mathrm{CNF}$.
5. ψ_x has $2^{d \lg n} C(q) = n^d C(q)$ clauses.

Final Formula

Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$
Let $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $x \in \{0,1\}^n$. This is the input to the PCP.
We have said how to take $\tau \in \{0,1\}^{d \lg n}$ and form ψ_τ.

1. ψ_τ is on $\le 2^q$ vars, a constant. Rewrite ψ_τ as a $3\mathrm{CNF}$.
2. ψ_τ has $\le C(q)$ clauses. Add clauses of the form $(x \vee x \vee x)$ with new vars x to get exactly $C(q)$ clauses.
3. Let ψ_x be the $\bigwedge$ of all the ψ_τ.
4. Note that ψ_x is $3\mathrm{CNF}$.
5. ψ_x has $2^{d \lg n} C(q) = n^d C(q)$ clauses.
6. Note that ψ_x is in 3CNF Form and has $C(q)n^d$ clauses.

Final Formula

Let $A \in \mathrm{NP}$ and $\epsilon > 0$. Then $\exists q, d \in \mathbb{N}$
Let $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $x \in \{0,1\}^n$. This is the input to the PCP.
We have said how to take $\tau \in \{0,1\}^{d \lg n}$ and form ψ_τ.

1. ψ_τ is on $\le 2^q$ vars, a constant. Rewrite ψ_τ as a $\mathrm{3CNF}$.
2. ψ_τ has $\le C(q)$ clauses. Add clauses of the form $(x \vee x \vee x)$ with new vars x to get exactly $C(q)$ clauses.
3. Let ψ_x be the $\bigwedge$ of all the ψ_τ.
4. Note that ψ_x is $\mathrm{3CNF}$.
5. ψ_x has $2^{d \lg n} C(q) = n^d C(q)$ clauses.
6. Note that ψ_x is in 3CNF Form and has $C(q)n^d$ clauses.

Going from x to ψ_x takes time poly in $|x| = n$.

MAX3SAT is Not PTAS: Set Up

Assume BWOC $(\forall \delta < 1)$ $\mathrm{MAX3SAT}$ is $(1-\delta)$-approximable.

MAX3SAT is Not PTAS: Set Up

Assume BWOC $(\forall \delta < 1)$ $\mathrm{MAX3SAT}$ is $(1-\delta)$-approximable.

We pick δ later. It will matter.

MAX3SAT is Not PTAS: Set Up

Assume BWOC $(\forall \delta < 1)$ $\mathrm{MAX3SAT}$ is $(1-\delta)$-approximable.

We pick δ later. It will matter.

We call the approx algorithm that achieves this **app-alg**.

MAX3SAT is Not PTAS: Set Up

Assume BWOC $(\forall \delta < 1)$ $\mathrm{MAX3SAT}$ is $(1-\delta)$-approximable.

We pick δ later. It will matter.

We call the approx algorithm that achieves this **app-alg**.

Let $A \in \mathrm{NP}$. We pick ϵ later. It won't matter.

MAX3SAT is Not PTAS: Set Up

Assume BWOC $(\forall\delta < 1)$ $\mathrm{MAX3SAT}$ is $(1-\delta)$-approximable.

We pick δ later. It will matter.

We call the approx algorithm that achieves this **app-alg**.

Let $A \in \mathrm{NP}$. We pick ϵ later. It won't matter.

By PCP Thm $(\exists d, q \in \mathbb{N})[A \in \mathrm{PCP}(q, d \lg n, \epsilon)]$.

MAX3SAT is Not PTAS: Set Up

Assume BWOC $(\forall \delta < 1)$ $\mathrm{MAX3SAT}$ is $(1-\delta)$-approximable.

We pick δ later. It will matter.

We call the approx algorithm that achieves this **app-alg**.

Let $A \in \mathrm{NP}$. We pick ϵ later. It won't matter.

By PCP Thm $(\exists d, q \in \mathbb{N})[A \in \mathrm{PCP}(q, d \lg n, \epsilon)]$.

If we run the PCP with oracle y we say $\mathbf{PCP}^{y}$.

MAX3SAT is Not PTAS: Set Up

Assume BWOC $(\forall\delta < 1)$ $\mathrm{MAX3SAT}$ is $(1-\delta)$-approximable.

We pick δ later. It will matter.

We call the approx algorithm that achieves this **app-alg**.

Let $A \in \mathrm{NP}$. We pick ϵ later. It won't matter.

By PCP Thm $(\exists d, q \in \mathbb{N})[A \in \mathrm{PCP}(q, d \lg n, \epsilon)]$.

If we run the PCP with oracle y we say $\mathbf{PCP}^{y}$.

We use **app-alg** and **the PCP** to obtain $A \in \mathrm{P}$.

MAX3SAT is Not PTAS: A in P Algorithm

MAX3SAT is Not PTAS: A in P Algorithm

1. Input x.

MAX3SAT is Not PTAS: A in P Algorithm

1. Input x.
2. Form the $3\mathrm{CNF}$ formula ψ_x.

MAX3SAT is Not PTAS: A in P Algorithm

1. Input x.
2. Form the $3\mathrm{CNF}$ formula ψ_x.
3. Apply the approx to ψ_x.

MAX3SAT is Not PTAS: A in P Algorithm

1. Input x.
2. Form the $3\mathrm{CNF}$ formula ψ_x.
3. Apply the approx to ψ_x.
4. We will pick ϵ, δ such that there is a gap between what the approx yields if $x \in A$ and if $x \notin A$. Details on next "few" slides.

MAX3SAT is Not PTAS: A in P Algorithm

1. Input x.
2. Form the $3\mathrm{CNF}$ formula ψ_x.
3. Apply the approx to ψ_x.
4. We will pick ϵ, δ such that there is a gap between what the approx yields if $x \in A$ and if $x \notin A$. Details on next "few" slides.

   We will then finish the algorithm.

MAX3SAT is Not PTAS: $x \in A$ Case

Assume $x \in A$.

MAX3SAT is Not PTAS: $x \in A$ Case

Assume $x \in A$.

Then there is an oracle y so that, **for all τ**, the PCP, with τ, and using y for answers, **accepts.**

MAX3SAT is Not PTAS: $x \in A$ Case

Assume $x \in A$.

Then there is an oracle y so that, **for all τ**, the PCP, with τ, and using y for answers, **accepts.**

Formally

$$(\exists y)(\forall \tau \in \{0,1\}^{d \lg n})[\mathrm{PCP}^y(x,\tau) \text{ ACCEPTS}].$$

MAX3SAT is Not PTAS: $x \in A$ Case

Assume $x \in A$.

Then there is an oracle y so that, **for all τ**, the PCP, with τ, and using y for answers, **accepts.**

Formally

$$(\exists y)(\forall \tau \in \{0,1\}^{d \lg n})[\mathrm{PCP}^{y}(x,\tau) \text{ ACCEPTS}].$$

Hence there is a way to satisfy all $n^d C(q)$ clauses of ψ_τ simul.
So $\mathrm{OPT}(\psi_x) = n^d C(q)$.

MAX3SAT is Not PTAS: $x \notin A$ Case

Assume $x \notin A$.

MAX3SAT is Not PTAS: $x \notin A$ Case

Assume $x \notin A$.

For all oracles y, for **at most ϵ of the τ**, the PCP, with τ, and using y for answers, **accepts.**

MAX3SAT is Not PTAS: $x \notin A$ Case

Assume $x \notin A$.

For all oracles y, for **at most ϵ of the τ**, the PCP, with τ, and using y for answers, **accepts.**

Formally

MAX3SAT is Not PTAS: $x \notin A$ Case

Assume $x \notin A$.

For all oracles y, for **at most ϵ of the τ**, the PCP, with τ, and using y for answers, **accepts.**

Formally

$(\forall y \in \{0,1\}^q)$

$$\text{For} \le \epsilon(2^{d \lg n}) \text{ of the } \tau \in \{0,1\}^{d \lg n} [\mathrm{PCP}^y(x, \tau) \text{ACCEPTS}].$$

If $x \notin A$ How Many Clauses Satisfied?

Let y be the oracle (Truth Assignment) that yields $\mathrm{OPT}(\psi_x)$

If $x \notin A$ How Many Clauses Satisfied?

Let y be the oracle (Truth Assignment) that yields $\mathrm{OPT}(\psi_x)$

$$\psi_x = \bigwedge \psi_\tau$$

If $x \notin A$ How Many Clauses Satisfied?

Let y be the oracle (Truth Assignment) that yields $\mathrm{OPT}(\psi_x)$

$$\psi_x = \bigwedge \psi_\tau$$

Recall Each ψ_x has exactly $C(q)$ clauses.

If $x \notin A$ How Many Clauses Satisfied?

Let y be the oracle (Truth Assignment) that yields $\mathrm{OPT}(\psi_x)$

$$\psi_x = \bigwedge \psi_\tau$$

Recall Each ψ_x has exactly $C(q)$ clauses.

At most ϵ of the τ's are satisfied.
Worst case For $\phi_\tau \notin \mathrm{SAT}$, $\mathrm{OPT}(\phi_\tau) = C(q) - 1$.

If $x \notin A$ How Many Clauses Satisfied?

Let y be the oracle (Truth Assignment) that yields $\mathrm{OPT}(\psi_x)$

$$\psi_x = \bigwedge \psi_\tau$$

Recall Each ψ_x has exactly $C(q)$ clauses.

At most ϵ of the τ's are satisfied.
Worst case For $\phi_\tau \notin \mathrm{SAT}$, $\mathrm{OPT}(\phi_\tau) = C(q) - 1$.

So Number of clauses satisfied is

$$\epsilon n^d C(q) + (1-\epsilon) n^d (C(q) - 1) = n^d(\epsilon C(q) + (1-\epsilon)(C(q)-1))$$

$$= n^d(\epsilon C(q) + C(q) - \epsilon C(q) - 1 + \epsilon) = n^d(C(q) - 1 + \epsilon)$$

Apply Approx and See What Happens

Apply Approx and See What Happens

$\boldsymbol{x} \in \boldsymbol{A}$ $\mathrm{MAX3SAT}(\psi_x) = n^d C(q)$, app-alg $\geq (1-\delta)n^d C(q)$.

Apply Approx and See What Happens

$\boldsymbol{x} \in \boldsymbol{A}$ $\mathrm{MAX3SAT}(\psi_x) = n^d C(q)$, app-alg $\geq (1-\delta)n^d C(q)$.

$\boldsymbol{x} \notin \boldsymbol{A}$ $\mathrm{MAX3SAT}(\psi_x) \leq n^d(C(q) - 1 + \epsilon)$, so app-alg
$\leq n^d(C(q) - 1 + \epsilon)$.

Apply Approx and See What Happens

$x \in A$ $\mathrm{MAX3SAT}(\psi_x) = n^d C(q)$, app-alg $\geq (1-\delta)n^d C(q)$.

$x \notin A$ $\mathrm{MAX3SAT}(\psi_x) \leq n^d(C(q) - 1 + \epsilon)$, so app-alg $\leq n^d(C(q) - 1 + \epsilon)$.

For Gap Need

$$n^d(C(q) - 1 + \epsilon) < (1-\delta)n^d C(q)$$

$$\delta < \frac{1-\epsilon}{C(q)}$$

We Won't Pick ϵ Cleverly

For Gap Need

$$\delta < \frac{1-\epsilon}{C(q)}$$

We Won't Pick ϵ Cleverly

For Gap Need

$$\delta < \frac{1-\epsilon}{C(q)}$$

We want to maximize δ.

We Won't Pick ϵ Cleverly

For Gap Need

$$\delta < \frac{1-\epsilon}{C(q)}$$

We want to maximize δ.

The smaller ϵ is, the bigger q is, so the bigger $C(q)$ is.

We Won't Pick ϵ Cleverly

For Gap Need

$$\delta < \frac{1-\epsilon}{C(q)}$$

We want to maximize δ.

The smaller ϵ is, the bigger q is, so the bigger $C(q)$ is.

If we knew how all of these related we would pick ϵ carefully to maximize $\frac{1-\epsilon}{C(q)}$.

We Won't Pick ϵ Cleverly

For Gap Need

$$\delta < \frac{1-\epsilon}{C(q)}$$

We want to maximize δ.

The smaller ϵ is, the bigger q is, so the bigger $C(q)$ is.

If we knew how all of these related we would pick ϵ carefully to maximize $\frac{1-\epsilon}{C(q)}$.
We don't.

We Won't Pick ϵ Cleverly

For Gap Need

$$\delta < \frac{1-\epsilon}{C(q)}$$

We want to maximize δ.

The smaller ϵ is, the bigger q is, so the bigger $C(q)$ is.

If we knew how all of these related we would pick ϵ carefully to maximize $\frac{1-\epsilon}{C(q)}$.
We don't.

But all we want is there is **some** δ so we can show MAX3SAT has no PTAS.

We Won't Pick ϵ Cleverly

For Gap Need

$$\delta < \frac{1-\epsilon}{C(q)}$$

We want to maximize δ.

The smaller ϵ is, the bigger q is, so the bigger $C(q)$ is.

If we knew how all of these related we would pick ϵ carefully to maximize $\frac{1-\epsilon}{C(q)}$.
We don't.

But all we want is there is **some** δ so we can show MAX3SAT has no PTAS.

We pick $\epsilon = \frac{1}{4}$, but still call it ϵ.

We Won't Pick ϵ Cleverly

For Gap Need

$$\delta < \frac{1-\epsilon}{C(q)}$$

We want to maximize δ.

The smaller ϵ is, the bigger q is, so the bigger $C(q)$ is.

If we knew how all of these related we would pick ϵ carefully to maximize $\frac{1-\epsilon}{C(q)}$.
We don't.

But all we want is there is **some** δ so we can show MAX3SAT has no PTAS.

We pick $\epsilon = \frac{1}{4}$, but still call it ϵ.

We pick $\delta = \frac{1-\epsilon}{2C(q)}$.

MAX3SAT is Not PTAS: A in P Algorithm

Let $\epsilon = \frac{1}{4}$. Let q, d be such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $C(q)$ be as discussed above.

MAX3SAT is Not PTAS: A in P Algorithm

Let $\epsilon = \frac{1}{4}$. Let q, d be such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $C(q)$ be as discussed above.

Let $\delta = \frac{1-\epsilon}{2C(q)}$.

MAX3SAT is Not PTAS: A in P Algorithm

Let $\epsilon = \frac{1}{4}$. Let q, d be such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $C(q)$ be as discussed above.

Let $\delta = \frac{1-\epsilon}{2C(q)}$.

We show there is no $(1-\delta)$-approx for MAX3SAT.
Assume, BWOC, that there is such a app-alg. We use the app-alg, and the PCP, to get $A \in \mathrm{P}$.

MAX3SAT is Not PTAS: A in P Algorithm

Let $\epsilon = \frac{1}{4}$. Let q, d be such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $C(q)$ be as discussed above.
Let $\delta = \frac{1-\epsilon}{2C(q)}$.
We show there is no $(1-\delta)$-approx for MAX3SAT.
Assume, BWOC, that there is such a app-alg. We use the app-alg, and the PCP, to get $A \in \mathrm{P}$.

1. Input x.

MAX3SAT is Not PTAS: A in P Algorithm

Let $\epsilon = \frac{1}{4}$. Let q, d be such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $C(q)$ be as discussed above.
Let $\delta = \frac{1-\epsilon}{2C(q)}$.
We show there is no $(1-\delta)$-approx for MAX3SAT.
Assume, BWOC, that there is such a app-alg. We use the app-alg, and the PCP, to get $A \in \mathrm{P}$.

1. Input x.
2. Form the $3\mathrm{CNF}$ formula ψ_x. Let X be the number of clauses.

MAX3SAT is Not PTAS: A in P Algorithm

Let $\epsilon = \frac{1}{4}$. Let q, d be such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $C(q)$ be as discussed above.
Let $\delta = \frac{1-\epsilon}{2C(q)}$.
We show there is no $(1-\delta)$-approx for MAX3SAT.
Assume, BWOC, that there is such a app-alg. We use the app-alg, and the PCP, to get $A \in \mathrm{P}$.

1. Input x.
2. Form the $\mathrm{3CNF}$ formula ψ_x. Let X be the number of clauses.
3. Apply the approx to ψ_x. Call the result Y.

MAX3SAT is Not PTAS: A in P Algorithm

Let $\epsilon = \frac{1}{4}$. Let q, d be such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $C(q)$ be as discussed above.
Let $\delta = \frac{1-\epsilon}{2C(q)}$.
We show there is no $(1-\delta)$-approx for MAX3SAT.
Assume, BWOC, that there is such a app-alg. We use the app-alg, and the PCP, to get $A \in \mathrm{P}$.

1. Input x.
2. Form the $3\mathrm{CNF}$ formula ψ_x. Let X be the number of clauses.
3. Apply the approx to ψ_x. Call the result Y.
4. If $Y \geq (1-\delta)n^d C(q)$ then output YES, $x \in A$.

MAX3SAT is Not PTAS: A in P Algorithm

Let $\epsilon = \frac{1}{4}$. Let q, d be such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $C(q)$ be as discussed above.
Let $\delta = \frac{1-\epsilon}{2C(q)}$.
We show there is no $(1-\delta)$-approx for MAX3SAT.
Assume, BWOC, that there is such a app-alg. We use the app-alg, and the PCP, to get $A \in \mathrm{P}$.

1. Input x.
2. Form the $3\mathrm{CNF}$ formula ψ_x. Let X be the number of clauses.
3. Apply the approx to ψ_x. Call the result Y.
4. If $Y \ge (1-\delta)n^d C(q)$ then output YES, $x \in A$.
5. If $Y \le n^d(C(q) - 1 + \epsilon)$ then output NO, $x \notin A$.

MAX3SAT is Not PTAS: A in P Algorithm

Let $\epsilon = \frac{1}{4}$. Let q, d be such that $A \in \mathrm{PCP}(q, d \lg n, \epsilon)$.
Let $C(q)$ be as discussed above.
Let $\delta = \frac{1-\epsilon}{2C(q)}$.
We show there is no $(1-\delta)$-approx for MAX3SAT.
Assume, BWOC, that there is such a app-alg. We use the app-alg, and the PCP, to get $A \in \mathrm{P}$.

1. Input x.
2. Form the $3\mathrm{CNF}$ formula ψ_x. Let X be the number of clauses.
3. Apply the approx to ψ_x. Call the result Y.
4. If $Y \geq (1-\delta)n^d C(q)$ then output YES, $x \in A$.
5. If $Y \leq n^d(C(q) - 1 + \epsilon)$ then output NO, $x \notin A$.

By the commentary in the last few slides, and the choice of δ, exactly one of the inequalities for Y holds.