BILL AND NATHAN, RECORD LECTURE!!!!

BILL RECORD LECTURE!!!
TSP cannot be Approximated Unless P=NP
TSP
In this slide packet $G$ is always a weighted graph with natural number weights
Recall TSP is the following problem

1. **Input** $G$ and $k \in \mathbb{N}$.
2. **Output** YES if there is a Ham Cycle in $G$ of weight $\leq k$, NO otherwise.

This is a Decision Problem which has a YES-NO answer. What we really want is to find the optimal Ham Cycle. Since TSP is NPC, finding the optimal is likely hard. But what about approximating it? Need to define this carefully.
Recall TSP is the following problem

1. **Input** $G$ and $k \in \mathbb{N}$.
Recall TSP is the following problem

1. **Input** $G$ and $k \in \mathbb{N}$.
2. **Output** YES if there is a Ham Cycle in $G$ of weight $\leq k$, NO otherwise.
Recall TSP is the following problem

1. **Input** $G$ and $k \in \mathbb{N}$.

2. **Output** YES if there is a Hamiltonian Cycle in $G$ of weight $\leq k$, NO otherwise.

This is a **Decision Problem** which has a YES-NO answer.
Recall TSP is the following problem

1. **Input** $G$ and $k \in \mathbb{N}$.
2. **Output** YES if there is a Ham Cycle in $G$ of weight $\leq k$, NO otherwise.

This is a **Decision Problem** which has a YES-NO answer.

What we really want is to **find** the optimal Ham Cycle.
**Recall** TSP is the following problem

1. **Input** $G$ and $k \in \mathbb{N}$.
2. **Output** YES if there is a Ham Cycle in $G$ of weight $\leq k$, NO otherwise.

This is a **Decision Problem** which has a YES-NO answer.

What we really want is to **find** the optimal Ham Cycle.

Since TSP is **NPC**, finding the optimal is likely hard.
TSP

**Recall** TSP is the following problem

1. **Input** $G$ and $k \in \mathbb{N}$.
2. **Output** YES if there is a Ham Cycle in $G$ of weight $\leq k$, NO otherwise.

This is a **Decision Problem** which has a YES-NO answer.

What we really want is to **find** the optimal Ham Cycle.

Since TSP is **NPC**, finding the optimal is likely hard.

But what about **approximating it**? Need to define this carefully.
An \( \alpha \)-Approx For TSP

**Def** \( \text{OPT}(G) \) is the weight of the lowest weight Ham Cycle of \( G \).
An \( \alpha \)-Approx For TSP

**Def** \( \text{OPT}(G) \) is the weight of the lowest weight Ham Cycle of \( G \). Clearly if finding \( \text{OPT}(G) \) is in \( P \) then \( P = NP \).
An $\alpha$-Approx For TSP

**Def** $\text{OPT}(G)$ is the weight of the lowest weight Ham Cycle of $G$. Clearly if finding $\text{OPT}(G)$ is in $P$ then $P = \text{NP}$.

**Def** Let $\alpha > 1$. An $\alpha$-approx for TSP is a poly time algorithm that, on input $G$, returns a cycle that is $\leq \alpha \text{OPT}(G)$.
An \( \alpha \)-Approx For TSP

**Def** OPT(\( G \)) is the weight of the lowest weight Ham Cycle of \( G \). Clearly if finding OPT(\( G \)) is in P then P = NP.

**Def** Let \( \alpha > 1 \). An \( \alpha \)-approx for TSP is a poly time algorithm that, on input \( G \), returns a cycle that is \( \leq \alpha \)OPT(\( G \)). What if we can get better and better approximations?
An $\alpha$-Approx For TSP

**Def** $OPT(G)$ is the weight of the lowest weight Ham Cycle of $G$. Clearly if finding $OPT(G)$ is in $P$ then $P = NP$.

**Def** Let $\alpha > 1$. An $\alpha$-approx for TSP is a poly time algorithm that, on input $G$, returns a cycle that is $\leq \alpha OPT(G)$.

What if we can get better and better approximations?

**Def** A Poly time Approx Scheme (PTAS) for TSP is a poly time algorithm that, on input $(G, \epsilon)$, returns a cycle that is $\leq (1 + \epsilon)OPT(G)$. Run time depends on $\epsilon$. Can be bad: $n^{2^{1/\epsilon^2}}$. 

VOTE assuming $P \neq NP$.

1) There is a PTAS for TSP.
2) There is an $\alpha$ such that (1) TSP has an $\alpha$-approx but (2) for all $\beta < \alpha$ there is no $\beta$-approx for TSP.
3) There is no such $\alpha$. E.g., there is no $(1 + 1/1000)$-approx for TSP.

ANSWER: 3, no approx. But there is approx for subcases.
An $\alpha$-Approx For TSP

**Def** $\text{OPT}(G)$ is the weight of the lowest weight Ham Cycle of $G$. Clearly if finding $\text{OPT}(G)$ is in $\mathbf{P}$ then $\mathbf{P} = \mathbf{NP}$.

**Def** Let $\alpha > 1$. An $\alpha$-approx for TSP is a poly time algorithm that, on input $G$, returns a cycle that is $\leq \alpha \text{OPT}(G)$.

What if we can get better and better approximations?

**Def** A Poly time Approx Scheme (PTAS) for TSP is a poly time algorithm that, on input $(G, \epsilon)$, returns a cycle that is $\leq (1 + \epsilon)\text{OPT}(G)$. Run time depends on $\epsilon$. Can be bad: $n^{21/\epsilon^2}$.

VOTE assuming $\mathbf{P} \neq \mathbf{NP}$.

1) There is a PTAS for TSP.
2) There is an $\alpha$ such that (1) TSP has an $\alpha$-approx but (2) for all $\beta < \alpha$ there is no $\beta$-approx for TSP.
3) There is no such $\alpha$. E.g., there is no $(1 + 1/2000)$-approx for TSP.

ANSWER: 3, no approx. But there is approx for subcases.
An $\alpha$-Approx For TSP

**Def** $\text{OPT}(G)$ is the weight of the lowest weight Ham Cycle of $G$. Clearly if finding $\text{OPT}(G)$ is in P then $P = NP$.

**Def** Let $\alpha > 1$. An $\alpha$-approx for TSP is a poly time algorithm that, on input $G$, returns a cycle that is $\leq \alpha \text{OPT}(G)$.

What if we can get better and better approximations?

**Def** A **Poly time Approx Scheme (PTAS)** for TSP is a poly time algorithm that, on input $(G, \epsilon)$, returns a cycle that is $\leq (1 + \epsilon) \text{OPT}(G)$. Run time depends on $\epsilon$. Can be bad: $n^{21/\epsilon^2}$.

VOTE assuming $P \neq NP$.

1) There is a PTAS for TSP.
**An $\alpha$-Approx For TSP**

**Def** $\text{OPT}(G)$ is the weight of the lowest weight Ham Cycle of $G$. Clearly if finding $\text{OPT}(G)$ is in P then $P = NP$.

**Def** Let $\alpha > 1$. An $\alpha$-approx for TSP is a poly time algorithm that, on input $G$, returns a cycle that is $\leq \alpha \text{OPT}(G)$.

What if we can get better and better approximations?

**Def** A *Poly time Approx Scheme (PTAS)* for TSP is a poly time algorithm that, on input $(G, \epsilon)$, returns a cycle that is $\leq (1 + \epsilon)\text{OPT}(G)$. Run time depends on $\epsilon$. Can be bad: $n^{21/\epsilon^2}$.

VOTE assuming $P \neq NP$.

1) There is a PTAS for TSP.

2) There is an $\alpha$ such that (1) TSP has an $\alpha$-approx but (2) for all $\beta < \alpha$ there is no $\beta$-approx for TSP.
An $\alpha$-Approx For TSP

**Def** \( \text{OPT}(G) \) is the weight of the lowest weight Ham Cycle of \( G \). Clearly if finding \( \text{OPT}(G) \) is in \( P \) then \( P = NP \).

**Def** Let \( \alpha > 1 \). An \( \alpha \)-approx for TSP is a poly time algorithm that, on input \( G \), returns a cycle that is \( \leq \alpha \text{OPT}(G) \).

What if we can get better and better approximations?

**Def** A Poly time Approx Scheme (PTAS) for TSP is a poly time algorithm that, on input \((G, \epsilon)\), returns a cycle that is \( \leq (1 + \epsilon)\text{OPT}(G) \). Run time depends on \( \epsilon \). Can be bad: \( n^{21/\epsilon^2} \).

VOTE assuming \( P \neq NP \).

1) There is a PTAS for TSP.

2) There is an \( \alpha \) such that (1) TSP has an \( \alpha \)-approx but (2) for all \( \beta < \alpha \) there is no \( \beta \)-approx for TSP.

3) There is no such \( \alpha \). E.g., there is no \((1 + \frac{1}{2^{1000}})\)-approx for TSP.
An $\alpha$-Approx For TSP

**Def** $\text{OPT}(G)$ is the weight of the lowest weight Ham Cycle of $G$. Clearly if finding $\text{OPT}(G)$ is in $P$ then $P = NP$.

**Def** Let $\alpha > 1$. An $\alpha$-approx for TSP is a poly time algorithm that, on input $G$, returns a cycle that is $\leq \alpha \text{OPT}(G)$.

What if we can get better and better approximations?

**Def** A Poly time Approx Scheme (PTAS) for TSP is a poly time algorithm that, on input $(G, \epsilon)$, returns a cycle that is $\leq (1 + \epsilon)\text{OPT}(G)$. Run time depends on $\epsilon$. Can be bad: $n^{21/\epsilon^2}$. VOTE assuming $P \neq NP$.

1) There is a PTAS for TSP.
2) There is an $\alpha$ such that (1) TSP has an $\alpha$-approx but (2) for all $\beta < \alpha$ there is no $\beta$-approx for TSP.
3) There is no such $\alpha$. E.g., there is no $(1 + \frac{1}{21000})$-approx for TSP.

ANSWER: 3, no approx. But there is approx for subcases.
Approximating TSP

1. Metric TSP: TSP problem restricted to weighted graphs that are symmetric and satisfy the triangle inequality:
   \[ w(x, y) + w(y, z) \geq w(x, z) \]. Christofides (1976) and Serdyukov (1978) give a \( 3/2 \)-approximation to metric TSP.

2. Karlan, Klein, Oveis-Gharan (2020) got the first improvement over \( 3/2 \)-approx: a \( (3/2 - \epsilon) \)-approx to the metric TSP (\( \epsilon < 10^{-36} \)).

3. Euclidean TSP: TSP problem when the graph is a set of points in the plane and the weights are the Euclidean distances. Arora (1998) and Mitchell (1999) showed that, for all \( \epsilon \), there is an \( (1 + \epsilon) \)-approximation in time \( O(n \log n) \) for \( d, \epsilon \).

4. Arora and Mitchell actually have an algorithm that works on \( n \) points in \( \mathbb{R}^d \) that runs in time \( O(n \log n \epsilon^{-d-1}) \).
Approximating TSP

1. **Metric TSP**: TSP problem restricted to weighted graphs that are symmetric and satisfy the triangle inequality: \( w(x, y) + w(y, z) \geq w(x, z) \). Christofides (1976) and Serdyukov (1978) gives a \( \frac{3}{2} \)-approximation to metric TSP.

2. Karlan, Klein, Oveis-Gharan (2020) got the first improvement over \( \frac{3}{2} \)-approx: a \( \left( \frac{3}{2} - \epsilon \right) \)-approx to the metric TSP (\( \epsilon < 10^{-36} \)).

3. **Euclidean TSP**: TSP problem when the graph is a set of points in the plane and the weights are the Euclidean distances. Arora (1998) and Mitchell (1999) showed that, for all \( \epsilon \), there is an \( (1 + \epsilon) \)-approximation in time \( O(n \cdot \log n \cdot \epsilon^{-O(1/\epsilon)}) \).

4. Arora and Mitchell actually have an algorithm that works on \( n \) points in \( \mathbb{R}^d \) that runs in time \( O(n \cdot \log n \cdot \epsilon^{-O(\sqrt{d}/\epsilon)})^{d-1} \).
Approximating TSP

1. **Metric TSP**: TSP problem restricted to weighted graphs that are symmetric and satisfy the triangle inequality: \( w(x, y) + w(y, z) \geq w(x, z) \). Christofides (1976) and Serdyukov (1978) gives a \( \frac{3}{2} \)-approximation to metric TSP.

2. Karlan, Klein, Oveis-Gharan (2020) got the first improvement over \( \frac{3}{2} \)-approx: a \( \left( \frac{3}{2} - \epsilon \right) \)-approx to the metric TSP \( (\epsilon < 10^{-36}) \).
Approximating TSP

1. **Metric TSP**: TSP problem restricted to weighted graphs that are symmetric and satisfy the triangle inequality: $w(x, y) + w(y, z) \geq w(x, z)$. Christofides (1976) and Serdyukov (1978) gives a $\frac{3}{2}$-approximation to metric TSP.

2. Karlan, Klein, Oveis-Gharan (2020) got the first improvement over $\frac{3}{2}$-approx: a $(\frac{3}{2} - \epsilon)$-approx to the metric TSP ($\epsilon < 10^{-36}$).

3. **Euclidean TSP**: TSP problem when the graph is a set of points in the plane and the weights are the Euclidean distances.
Approximating TSP

1. **Metric TSP**: TSP problem restricted to weighted graphs that are symmetric and satisfy the triangle inequality: \( w(x, y) + w(y, z) \geq w(x, z) \). Christofides (1976) and Serdyukov (1978) gives a \( \frac{3}{2} \)-approximation to metric TSP.

2. Karlan, Klein, Oveis-Gharan (2020) got the first improvement over \( \frac{3}{2} \)-approx: a \((\frac{3}{2} - \epsilon)\)-approx to the metric TSP \((\epsilon < 10^{-36})\).

3. **Euclidean TSP**: TSP problem when the graph is a set of points in the plane and the weights are the Euclidean distances. Arora (1998) and Mitchell (1999) showed that, for all \( \epsilon \), there is an \((1 + \epsilon)\)-approximation in time \( O(n(\log n)^{O(1/\epsilon)}) \).
Approximating TSP

1. **Metric TSP**: TSP problem restricted to weighted graphs that are symmetric and satisfy the triangle inequality:
   \[ w(x, y) + w(y, z) \geq w(x, z) \]. Christofides (1976) and Serdyukov (1978) gives a \( \frac{3}{2} \)-approximation to metric TSP.

2. Karlan, Klein, Oveis-Gharan (2020) got the first improvement over \( \frac{3}{2} \)-approx: a \( (\frac{3}{2} - \epsilon) \)-approx to the metric TSP (\( \epsilon < 10^{-36} \)).

3. **Euclidean TSP**: TSP problem when the graph is a set of points in the plane and the weights are the Euclidean distances. Arora (1998) and Mitchell (1999) showed that, for all \( \epsilon \), there is an \( (1 + \epsilon) \)-approximation in time \( O(n(\log n)^{O(1/\epsilon)}) \).

4. Arora and Mitchell actually have an algorithm that works on \( n \) points in \( \mathbb{R}^d \) that runs in time \( O(n(\log n)^{O(\sqrt{d}/\epsilon)^{d-1}}}) \).
TSP Does Not have an $\alpha$-Approx
If TSP has an approx then HAMC is in P

Assume TSP has an $\alpha$-approx via Program $M$. $\alpha > 1$. 
If TSP has an approx then HAMC is in P

Assume TSP has an $\alpha$-approx via Program $M$. $\alpha > 1$.
1) Input $G$, an unweighted Graph on $n$ vertices.
If TSP has an approx then HAMC is in P

Assume TSP has an $\alpha$-approx via Program $M$. $\alpha > 1$.
1) Input $G$, an unweighted Graph on $n$ vertices.
2) Let $G'$ be the weighed graph where every edge in $G$ has weight 1 and every non-edge has weight $B$ where we determine $B$ later.
If TSP has an approx then HAMC is in P

Assume TSP has an $\alpha$-approx via Program $M$. $\alpha > 1$.

1) Input $G$, an unweighted Graph on $n$ vertices.
2) Let $G'$ be the weighed graph where every edge in $G$ has weight 1 and every non-edge has weight $B$ where we determine $B$ later.

Comment
If $G$ has a HAMC then $\text{OPT}(G') \leq n$. 
If TSP has an approx then HAMC is in P

Assume TSP has an $\alpha$-approx via Program $M$. $\alpha > 1$.

1) Input $G$, an unweighted Graph on $n$ vertices.
2) Let $G'$ be the weighed graph where every edge in $G$ has weight 1 and every non-edge has weight $B$ where we determine $B$ later.

Comment
If $G$ has a HAMC then $\text{OPT}(G') \leq n$.
If $G$ has no HAMC then $\text{OPT}(G') \geq B$.
If TSP has an approx then HAMC is in P

Assume TSP has an $\alpha$-approx via Program $M$. $\alpha > 1$.
1) Input $G$, an unweighted Graph on $n$ vertices.
2) Let $G'$ be the weighed graph where every edge in $G$ has weight 1 and every non-edge has weight $B$ where we determine $B$ later.

Comment
If $G$ has a HAMC then $\text{OPT}(G') \leq n$.
If $G$ has no HAMC then $\text{OPT}(G') \geq B$.
3) Run the $\alpha$-approx on $G'$. 
If TSP has an approx then HAMC is in P

Assume TSP has an \( \alpha \)-approx via Program \( M \). \( \alpha > 1 \).

1) Input \( G \), an unweighted Graph on \( n \) vertices.
2) Let \( G' \) be the weighed graph where every edge in \( G \) has weight 1 and every non-edge has weight \( B \) where we determine \( B \) later.

**Comment**

If \( G \) has a HAMC then \( \text{OPT}(G') \leq n \).

If \( G \) has no HAMC then \( \text{OPT}(G') \geq B \).

3) Run the \( \alpha \)-approx on \( G' \).

**Comment**

If \( G \) has a HAMC then \( \text{OPT}(G') \leq n \) so \( M(G') \leq \alpha n \).
If TSP has an approx then HAMC is in P

Assume TSP has an \( \alpha \)-approx via Program \( M \). \( \alpha > 1 \).
1) Input \( G \), an unweighted Graph on \( n \) vertices.
2) Let \( G' \) be the weighed graph where every edge in \( G \) has weight 1 and every non-edge has weight \( B \) where we determine \( B \) later.

**Comment**
If \( G \) has a HAMC then \( \text{OPT}(G') \leq n \).
If \( G \) has no HAMC then \( \text{OPT}(G') \geq B \).
3) Run the \( \alpha \)-approx on \( G' \).

**Comment**
If \( G \) has a HAMC then \( \text{OPT}(G') \leq n \) so \( M(G') \leq \alpha n \).
If \( G \) has no HAMC then \( \text{OPT}(G') \geq B \) so \( M(G') \geq B \).
If TSP has an approx then HAMC is in P

Assume TSP has an $\alpha$-approx via Program $M$. $\alpha > 1$.

1) Input $G$, an unweighted Graph on $n$ vertices.
2) Let $G'$ be the weighed graph where every edge in $G$ has weight 1 and every non-edge has weight $B$ where we determine $B$ later.

Comment
If $G$ has a HAMC then $\text{OPT}(G') \leq n$.
If $G$ has no HAMC then $\text{OPT}(G') \geq B$.
3) Run the $\alpha$-approx on $G'$.

Comment
If $G$ has a HAMC then $\text{OPT}(G') \leq n$ so $M(G') \leq \alpha n$.
If $G$ has no HAMC then $\text{OPT}(G') \geq B$ so $M(G') \geq B$.

Need to set $B$ such that $\alpha n < B$. $B = n^2$ will suffice.
If TSP has an approx then HAMC is in P

Assume TSP has an $\alpha$-approx via Program $M$. $\alpha > 1$.

1) Input $G$, an unweighted Graph on $n$ vertices.
2) Let $G'$ be the weighed graph where every edge in $G$ has weight
1 and every non-edge has weight $B$ where we determine $B$ later.

Comment
If $G$ has a HAMC then $\text{OPT}(G') \leq n$.
If $G$ has no HAMC then $\text{OPT}(G') \geq B$.
3) Run the $\alpha$-approx on $G'$.

Comment
If $G$ has a HAMC then $\text{OPT}(G') \leq n$ so $M(G') \leq \alpha n$.
If $G$ has no HAMC then $\text{OPT}(G') \geq B$ so $M(G') \geq B$.

Need to set $B$ such that $\alpha n < B$. $B = n^2$ will suffice.
4) Case 1: If $M(G') \leq \alpha n$ then output YES.
If TSP has an approx then HAMC is in P

Assume TSP has an $\alpha$-approx via Program $M$. $\alpha > 1$.
1) Input $G$, an unweighted Graph on $n$ vertices.
2) Let $G'$ be the weighed graph where every edge in $G$ has weight 1 and every non-edge has weight $B$ where we determine $B$ later.

Comment
If $G$ has a HAMC then $\text{OPT}(G') \leq n$.
If $G$ has no HAMC then $\text{OPT}(G') \geq B$.
3) Run the $\alpha$-approx on $G'$.

Comment
If $G$ has a HAMC then $\text{OPT}(G') \leq n$ so $M(G') \leq \alpha n$.
If $G$ has no HAMC then $\text{OPT}(G') \geq B$ so $M(G') \geq B$.
Need to set $B$ such that $\alpha n < B$. $B = n^2$ will suffice.
4) 
Case 1: If $M(G') \leq \alpha n$ then output YES.
Case 2: If $M(G') \geq B$ then output NO.
We can Do Better

We showed:

**Thm** Let $\alpha \geq 1$. If there is an $\alpha$-approx for TSP then $P=NP$. 
We can Do Better

We showed:

**Thm** Let $\alpha \geq 1$. If there is an $\alpha$-approx for TSP then $P=NP$.

If you look at the proof more carefully you can prove this:

**Thm** Let $\alpha(n)$ be a polynomial. If there is an $\alpha(n)$-approx for TSP then $P=NP$. 
History: 1971-1997

1. The TSP result goes back before 1978 and is folklore.
2. Before 1990 there were a few other non-approx results: Ind
set, Coloring, Knapsack, Prob others. All had elementary
though clever proofs, like the TSP result.
3. In 1991 a paper came out that showed:
   3.1 Many results like:
   \[ f \text{ has a PTAS IFF } g \text{ has a PTAS}. \]
   3.2 A class MAXSNP of functions that seemed to not have PTAS
   was defined.
   3.3 The problem:
\[ \text{MAX3SAT}(\varphi) = \max \text{ numb of clauses that can be satisfied} \]
   was shown complete for MAXSNP.
   But this was not very satisfying: it is plausible all these
   problems in MAXSNP had a PTAS.
1. The TSP result goes back before 1978 and is folklore.
1. The TSP result goes back before 1978 and is folklore.
2. Before 1990 there were a few other non-approx results: Ind set, Coloring, Knapsack, Prob others. All had elementary though clever proofs, like the TSP result.
1. The TSP result goes back before 1978 and is folklore.
2. Before 1990 there were a few other non-approx results: Ind set, Coloring, Knapsack, Prob others. All had elementary though clever proofs, like the TSP result.
3. In 1991 a paper came out that showed:

   3.1 Many results like: $f$ has a PTAS IFF $g$ has a PTAS.
   3.2 A class MAXSNP of functions that seemed to not have PTAS was defined.
   3.3 The problem: $\text{MAX3SAT}(\phi) = \text{max numb of clauses that can be satisfied}$ was shown complete for MAXSNP. But this was not very satisfying: it is plausible all these problems in MAXSNP had a PTAS.
1. The TSP result goes back before 1978 and is folklore.
2. Before 1990 there were a few other non-approx results: Ind set, Coloring, Knapsack, Prob others. All had elementary though clever proofs, like the TSP result.
3. In 1991 a paper came out that showed:
   3.1 Many results like: $f$ has a PTAS IFF $g$ has a PTAS.
1. The TSP result goes back before 1978 and is folklore.
2. Before 1990 there were a few other non-approx results: Ind set, Coloring, Knapsack, Prob others. All had elementary though clever proofs, like the TSP result.
3. In 1991 a paper came out that showed:
   3.1 Many results like: \( f \) has a PTAS IFF \( g \) has a PTAS.
   3.2 A class MAXSNP of functions that seemed to not have PTAS was defined.
**History: 1971-1997**

1. The TSP result goes back before 1978 and is folklore.
2. Before 1990 there were a few other non-approx results: Ind set, Coloring, Knapsack, Prob others. All had elementary though clever proofs, like the TSP result.
3. In 1991 a paper came out that showed:
   3.1 Many results like: \( f \) has a PTAS IFF \( g \) has a PTAS.
   3.2 A class MAXSNP of functions that seemed to not have PTAS was defined.
   3.3 The problem:
      \( \text{MAX3SAT}(\phi) = \text{max numb of clauses that can be satisfied} \)
        was shown complete for MAXSNP.
1. The TSP result goes back before 1978 and is folklore.
2. Before 1990 there were a few other non-approx results: Ind set, Coloring, Knapsack, Prob others. All had elementary though clever proofs, like the TSP result.
3. In 1991 a paper came out that showed:
   3.1 Many results like: \( f \) has a PTAS IFF \( g \) has a PTAS.
   3.2 A class MAXSNP of functions that seemed to not have PTAS was defined.
3.3 The problem:
   \( \text{MAX3SAT}(\phi) = \text{max numb of clauses that can be satisfied} \)
   was shown complete for MAXSNP.
   But this was not very satisfying: it is plausible all these problems in MAXSNP had a PTAS.
1. Motivated by (among other things) trying to find lower bounds on approx, the class \( \text{PCP}(q(n), r(n), \epsilon(n)) \) was defined.

2. In 1998 it was shown that \( \text{NP} = \text{PCP}(O(1), O(\log n), 1/n) \).
   This implied (with a lot of additional work):
   
   2.1 If MAX3SAT has a PTAS then \( \text{P} = \text{NP} \).
   
   2.2 If CLIQ can be well approximated then \( \text{P} = \text{NP} \).
   
   2.3 If SET COVER has an \((1 - o(1)) \ln(n)\) approx then \( \text{P} = \text{NP} \).

   (It is known to have a \( \ln(n) \)-approx. This took about 10 papers with many intermediary results.)
1. Motivated by (among other things) trying to find lower bounds on approx, the class $\text{PCP}(q(n), r(n)\epsilon(n))$ was defined.
1. Motivated by (among other things) trying to find lower bounds on approx, the class $\text{PCP}(q(n), r(n)\epsilon(n))$ was defined.

2. In 1998 it was shown that $\text{NP} = \text{PCP}(O(1), O(\log n), \frac{1}{n})$. This implied (with a lot of additional work):

2.1 If MAX3SAT has a PTAS then $P = \text{NP}$.

2.2 If CLIQ can be well approximated then $P = \text{NP}$.

2.3 If SET COVER has an $(1 - o(1)) \ln(n)$ approx then $P = \text{NP}$.

(It is known to have a $\ln(n)$-approx. This took about 10 papers with many intermediary results.)
1. Motivated by (among other things) trying to find lower bounds on approx, the class $\text{PCP}(q(n), r(n)\epsilon(n))$ was defined.

2. In 1998 it was shown that $\text{NP} = \text{PCP}(O(1), O(\log n), \frac{1}{n})$. This implied (with a lot of additional work):
   
   2.1 If MAX3SAT has a PTAS then $P = \text{NP}$.
1. Motivated by (among other things) trying to find lower bounds on approx, the class $\text{PCP}(q(n), r(n)\epsilon(n))$ was defined.

2. In 1998 it was shown that $\text{NP} = \text{PCP}(O(1), O(\log n), \frac{1}{n})$. This implied (with a lot of additional work):
   
   2.1 If MAX3SAT has a PTAS then $P = \text{NP}$.
   
   2.2 If CLIQ can be well approximated then $P = \text{NP}$.
1. Motivated by (among other things) trying to find lower bounds on approx, the class PCP\((q(n), r(n)\epsilon(n))\) was defined.

2. In 1998 it was shown that \(NP = PCP(O(1), O(\log n), \frac{1}{n})\). This implied (with a lot of additional work):
   2.1 If MAX3SAT has a PTAS then \(P = NP\).
   2.2 If CLIQ can be well approximated then \(P = NP\).
   2.3 If SET COVER has an \((1 - o(1))\ln(n)\) approx then \(P = NP\). (It is known to have a \(\ln(n)\)-approx. This took about 10 papers with many intermediary results.)