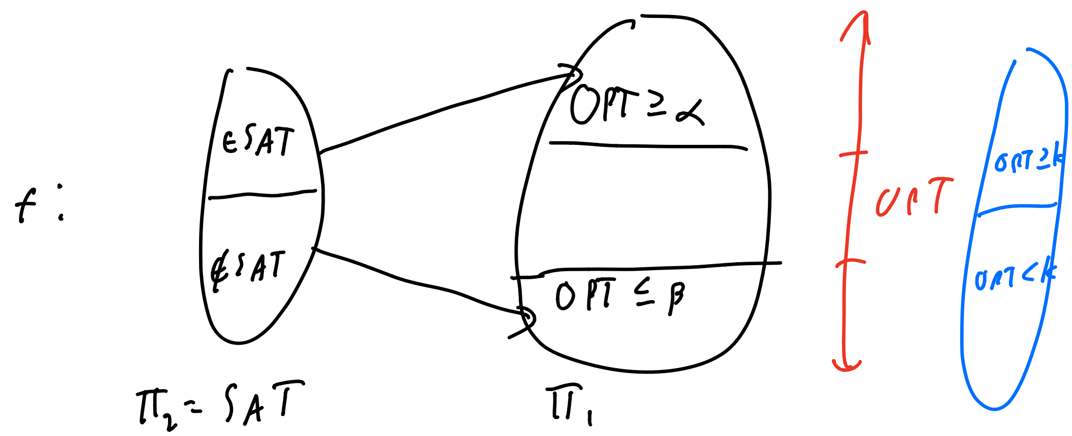
Hardness of Approximation:

Want to prove some maximization problem Π_1 hard to approx.

Gap reduction from some problem Π_2 we know is NP-hard (e.g., SAT)



Sps had γ-approx alg for Π_1 with $\gamma > \frac{\beta}{\alpha}$

$\Rightarrow$ on SAT instance x, run γ-approx on $f(x)$, get Π_1-solution of value ALG

$\Rightarrow$ If x not satisfiable, $ALG \le OPT(f(x)) \le \beta$

If x is satisfiable,

$$ALG \ge \gamma \cdot OPT(f(x)) > \frac{\beta}{\alpha} \cdot \alpha = \beta$$

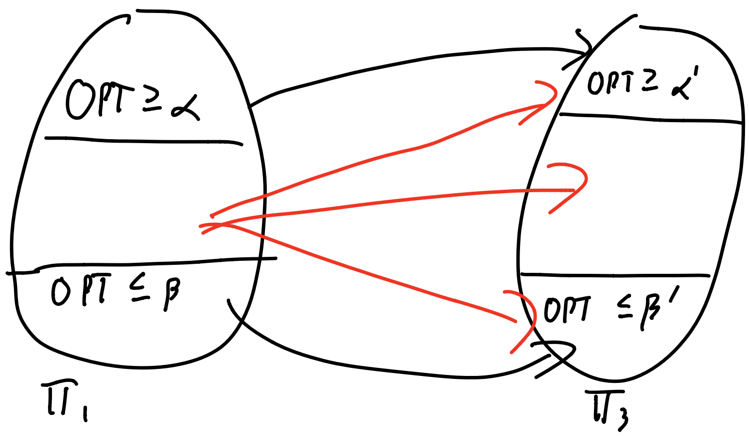
$\Rightarrow$ Poly time alg for SAT!

"Hard to distinguish $OPT \ge \alpha$ from $OPT \le \beta$"

Now suppose want to prove Π_3 hard to approximate.

Start with Π_1:

g:

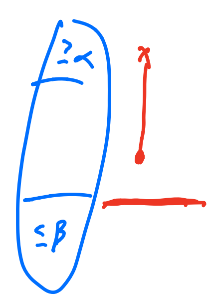


$\Rightarrow$ hard to approximate Π_3 better than $\frac{\beta'}{\alpha'}$

Doesn't matter what g does to middle instances!

Generic reduction framework to prove Π_3 hard to approx:

- Start with problem Π_1 where:
  - instances of Π_1 partitioned into YES, NO, MAYBE
  - it is NP-hard to distinguish YES instances from NO instances
- Design a reduction $f: \Pi_1 \to \Pi_3$ s.t.
  - completeness: If $x \in YES$, $OPT(f(x)) \geq \alpha$
  - Soundness: If $x \in NO$, $OPT(f(x)) \leq \beta$



$\Rightarrow$ β/α - hardness of approximation

Can get pretty far with this: Book 16.1, 16.2

Breakthrough: PCP Theorem. Digression into complexity theory.

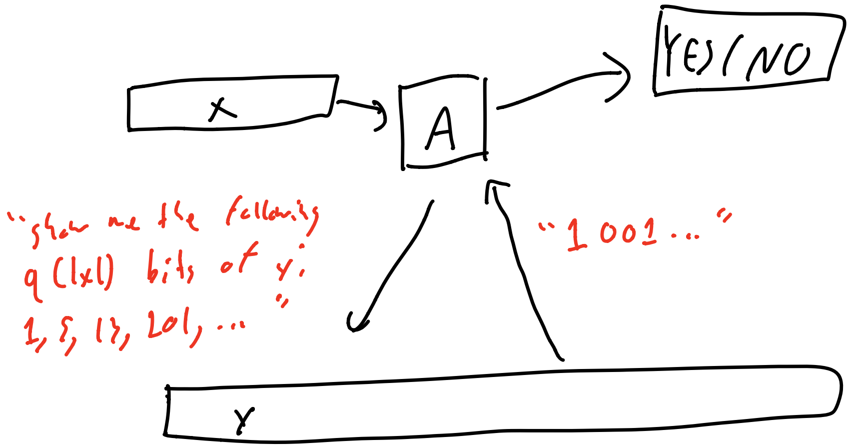
Def: $L \in NP$ if $\exists$ polynomial p and algorithm A s.t.

1) If $x \in L$, $\exists$ "proof" y s.t. $|y| \leq p(|x|)$ and $A(x,y)$ returns YES in time at most $p(|x|)$

2) If $x \notin L$, then $\forall y$, $A(x,y)$ returns NO

Def: A probabilistic proof system for L is a "verifier algorithm" A s.t. $\forall x$

1) A uses $r(|x|)$ random bits
2) A reads $q(|x|)$ bits of a "proof" y (nonadaptively)
3) If $x \in L$, then $\exists$ "proof" y s.t. A returns YES with probability $\geq c(|x|)$ (completeness)
4) If $x \notin L$, then $\forall y$, A returns YES with probability $\leq s(|x|)$ (soundness)



<u>Ex</u>: If $L \in NP$, then L has a probabilistic proof system with $r(n) = 0$, $q(n) = poly(n)$, $c(n) = 1$, $s(n) = 0$

<u>Def</u>: $PCP_{c(n), s(n)}(r(n), q(n))$ is the class of languages that have a probabilistic proof system with parameters $r(n), q(n), c(n), s(n)$

So $NP = PCP_{1,0}(0, poly(n))$

<u>Thm</u> [PCP Theorem]: $NP = PCP_{1, 1/2}(O(\log n), O(1))$

[AS'98, ALMSS'98]

Easy direction: $PCP_{1,1/2}(O(\log n), O(1)) \subseteq NP$

Pf: Let $L \in PCP_{1,1/2}(O(\log n), O(1))$

$\Rightarrow$ For each choice of $O(\log n)$ random bits, verifier checks $O(1)$ bits of proof

$\Rightarrow$ Only $2^{O(\log n)} \cdot O(1) = poly(n)$ bits of proof might ever be looked at

NP verifier A:

- Try all $2^{O(\log n)} = poly(n)$ choices of random bits, simulate PCP verifier on each
- Return YES if PCP verifier always returns YES
- Otherwise return NO

If $x \in L$, PCP verifier always returns YES $\Rightarrow$ A returns YES

If $x \notin L$, PCP verifier returns YES with probability $\leq \frac{1}{2}$

$\Rightarrow$ For at least on choice of random bits, returns NO

$\Rightarrow$ A returns NO

Hard direction: $NP \subseteq PCP_{1,1/2}(O(\log n), O(1))$

Back to approximation algorithms: why do we care about PCP Theorem?

Let Π arbitrary NP-complete problem (e.g., SAT)

$\Rightarrow$ by PCP thm, $\exists$ verifier with $O(\log n)$ random bits

$O(1)$ queries

$c(n) = 1$

$s(n) = 1/2$

New problem Π': given instance x of Π, find "proof" y maximizing $\Pr[\text{verifier accepts}]$

<u>Lemma</u>: Can't approximate Π' better than $1/2$!

<u>Pf</u>: Sps had $\gamma > \frac{1}{2}$-approx for Π'

$\Rightarrow$ if x YES of Π: $OPT(x) = 1 \Rightarrow ALG(x) \geq \gamma > \frac{1}{2}$

if x NO of Π: $OPT(x) \leq 1/2 \Rightarrow ALG(x) \leq \frac{1}{2}$

So run γ-approx to get y,

check all $2^{O(\log n)} = O(\text{poly}(n))$ possible queries,

if PCP verifier would accept on $> \frac{1}{2}$ of them, x YES

else x NO.

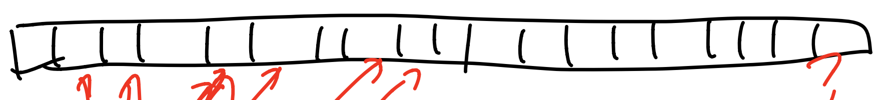
$\Rightarrow$ Could solve Π

More detailed: Π' is a CSP (constraint satisfaction problem)

For each of the poly(n) choices of random bits, verifier queries $O(1)$ spots of proof.

Query: some deterministic $f(\underbrace{y_1, y_2, \dots, y_k}_{\text{proof bits queried}}) \mapsto \{YES, NO\}$

proof y



Choice of random bits:

1: $f(y_2, y_5, y_8) = 1 / 0$

2: $f(y_3, y_7, y_{20}) = 1 / 0$

3: $f(y_3, y_5, y_9) = 1 / 0$

Maximizing Pr[verifier accepts y]: Choose bits of y to maximize # satisfied constraints

Completeness $c(n)$, soundness $s(n)$ $\Rightarrow$ hardness of $\frac{s(n)}{c(n)}$

Can rewrite arbitrary constraints as 3CNF formulae

$\Rightarrow$ hardness of $\frac{15}{16}$ for Max-3SAT

Can do even better through other versions of PCP Thm

Def: $\text{odd}(x_1,x_2,x_3) = \begin{cases} 1 & \text{if } x_1+x_2+x_3 \text{ odd} \\ 0 & \text{otherwise} \end{cases}$

$\text{even}(x_1,x_2,x_3) = \begin{cases} 1 & \text{if } x_1+x_2+x_3 \text{ even} \\ 0 & \text{otherwise} \end{cases}$

Thm [Håstad]: For any constant $\varepsilon > 0$,

$NP \subseteq PCP_{1-\varepsilon, \frac{1}{2}+\varepsilon}(O(\log n), 3)$

and verifier restricted to odd, even functions

Thm: $\forall$ constant $\varepsilon > 0$, it is NP-hard to approximate Max-3SAT better than $\frac{7}{8}+\varepsilon$

Pf: Start with arbitrary NP-complete problem (e.g., SAT)

Let φ instance of SAT

By Håstad, $\exists$ verifier with $c(n) = 1-\varepsilon$, $s(n) = \frac{1}{2}+\varepsilon$,

$O(\log n)$ random bits,
3 queries,
even/odd tests only

Let $N = 2^{O(\log n)} = \mathrm{poly}(n)$ be # distinct random strings used

$\Rightarrow$ for each of N random strings, 3 bits and even/odd test

For each $\mathrm{odd}(x_i, x_j, x_k)$ test:

$$\begin{aligned} &x_i \vee x_j \vee x_k \\ &\overline{x_i} \vee \overline{x_j} \vee x_k \\ &\overline{x_i} \vee x_j \vee \overline{x_k} \\ &x_i \vee \overline{x_j} \vee \overline{x_k} \end{aligned} \quad \Rightarrow \quad \begin{array}{l} \text{If } \mathrm{odd}(x_i, x_j, x_k) = 1, \text{ all 4 satisfied} \\ \text{else exactly 3 satisfied} \end{array}$$

For each $\mathrm{even}(x_i, x_j, x_k)$ test:

$$\begin{aligned} &\overline{x_i} \vee x_j \vee x_k \\ &x_i \vee \overline{x_j} \vee x_k \\ &x_i \vee x_j \vee \overline{x_k} \\ &\overline{x_i} \vee \overline{x_j} \vee \overline{x_k} \end{aligned} \quad \Rightarrow \quad \begin{array}{l} \text{If } \mathrm{even}(x_i, x_j, x_k) = 1, \text{ all 4 satisfied} \\ \text{else exactly 3 satisfied} \end{array}$$

If $\varphi \in SAT$ (YES instance) $\Rightarrow \exists$ proof s.t. verifier accepts with prob. $\geq 1-\varepsilon$

$\Rightarrow$ can satisfy $\geq (1-\varepsilon)N$ of the even/odd constraints

$\Rightarrow$ can satisfy $\geq 4(1-\varepsilon)N + 3\varepsilon N = (4-\varepsilon)N$ clauses

If $\varphi \notin SAT$ (NO instance) $\Rightarrow \forall$ proofs, verifier accepts with prob. $\leq \frac{1}{2}+\varepsilon$

$\Rightarrow$ can satisfy $\leq 4(\frac{1}{2}+\varepsilon)N + 3(\frac{1}{2}-\varepsilon)N = (\frac{7}{2}+\varepsilon)N$ clauses

$\Rightarrow$ hardness of $\dfrac{(\frac{7}{2}+\varepsilon)N}{(4-\varepsilon)N} = \dfrac{7}{8}+\varepsilon'$