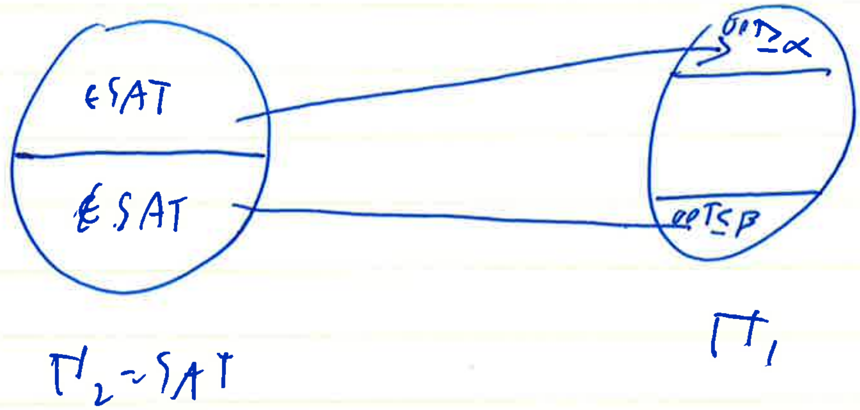
4/30/19

Class Notes 4/28/15: Inapproximability/maximization

Sps want to prove problem Π_1 hard to approximate. Gap reduction from a problem Π_2 that we already know is NP-hard (e.g. SAT)

f:



~~Sps could approximate Π_1 to better than $\frac{\alpha}{\beta}$.~~

Sps γ-approx to Π_1 w/ γ $\cancel{< \frac{\alpha}{\beta}}$ $> \frac{\beta}{\alpha}$

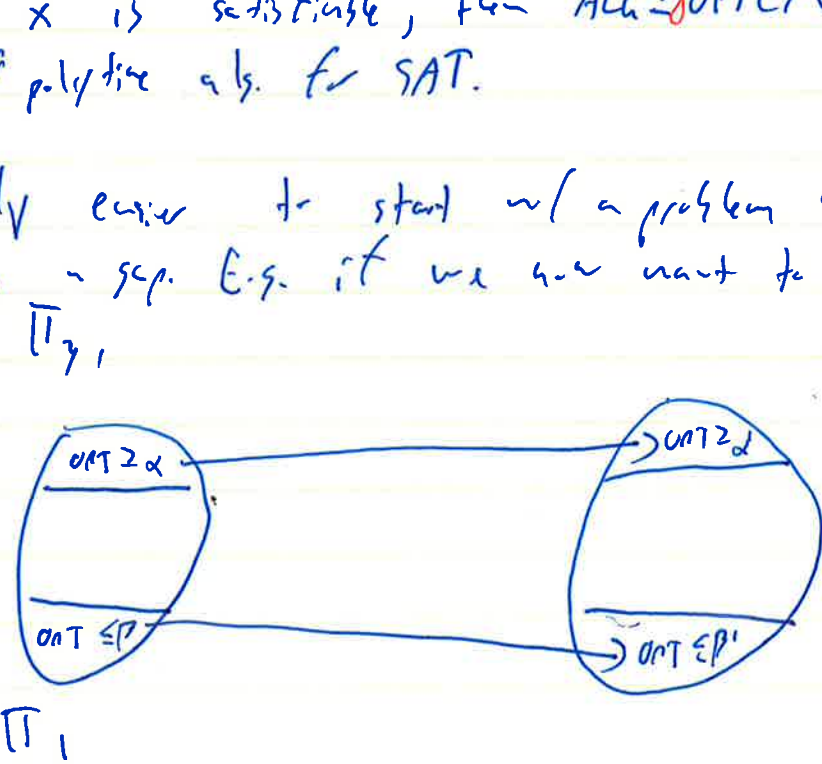
$\Rightarrow$ on SAT instance x, use reduction to get $f(x)$, run γ-approx on $f(x)$ to get ALG

If x not satisfiable, then $ALG \leq \cancel{\gamma \cdot} OPT(f(x)) \leq \cancel{\gamma \cdot} \beta \cancel{< \alpha}$

If x is satisfiable, then $ALG \geq \gamma OPT(f(x)) \geq \cancel{\alpha}\, \gamma\alpha > \beta$

$\Rightarrow$ poly time alg. for SAT.

Usually easier to start w/ a problem where there's already a gap. E.g. if we now want to prove hardness $\frac{\beta'}{\alpha'}$ $\cancel{\frac{\alpha'}{\beta'}}$ for Π_3,



Doesn't matter what reduction does to middle!

So generic reduction technique: start w/ problem Π_2 where ~~know NP-hard to distinguish~~ instances are partitioned into 3 groups: YES, NO, MAYBE, and is NP-hard to distinguish YES instances from NO instances. Showing Π_1 is $\frac{\alpha}{\beta}$-hard to approximate: reduction $f: \Pi_2 \to \Pi_1$ s.t.

- completeness: If $x \in YES$, $OPT(f(x)) \geq \alpha$
- soundness: If $x \in NO$, $OPT(f(x)) \leq \beta$

More powerful than usual reductions, since don't need to worry about what happens to MAYBE.

Can get pretty far w/ this (Book 16.1, 16.2), but real improvement from <u>PCP theorem</u>.

<u>Def</u>: $L \in NP$ if $\exists$ polynomial p and alg. A s.t. ~~if $x \in L$, there exists "proof" y s.t.~~

1) if $x \in L$, $\exists$ "proof" y s.t. $|y| \leq p(|x|)$ and $A(x,y)$ returns YES in time ~~$O(p(|x|))$~~ $\leq p(|x|)$
2) if $x \notin L$, then $\forall y$ $A(x,y)$ returns NO.

<u>Def</u>: A <u>probabilistic proof system</u> for L is a verifier alg. s.t.

1) Verifier uses $r(n)$ random bits $n = |x|$
2) Verifier reads $q(n)$ bits of a proof
3) If $x \in L$, then $\exists$ proof s.t. verifier returns YES w.p. $\geq c(n)$ (completeness)
4) If $x \notin L$, then $\forall y$ verifier returns YES w.p. $\leq s(n)$ (soundness)

Nonadaptive: makes all queries before getting any results

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

Def: $PCP_{c(n),s(n)}(r(n), q(n))$ is class of languages that have a probabilistic proof system w/ parameters $r(n)$, $q(n)$, $c(n)$, $s(n)$.

So in this notation, $NP = PCP_{1,0}(0, poly(n))$.

<u>Thm</u> (PCP Thm): $NP = PCP_{1,\frac{1}{2}}(O(\log n), O(1))$
[AS '98, ALMSS '98]

Easy direction: $PCP_{1,\frac{1}{2}}(O(\log n), O(1)) \subseteq NP$.
Sps $L \in PCP_{1,\frac{1}{2}}(O(\log n), O(1))$.
For each choice of $O(\log n)$ random bits, the verifier checks $O(1)$ bits of the proof
$\Rightarrow$ Only $2^{O(\log n)} \cdot O(1) = poly(n)$ bits of proof might ever be looked at.
So NP verifier A can simply try ~~each~~ all $2^{O(\log n)} = poly(n)$ choices of random bits, simulate PCP verifier on each one, return YES if all runs return YES, & return NO o/w
- if $x \in L$ then PCP ~~proof~~ verifier always returns YES
- if $x \notin L$ then by on $\geq \frac{1}{2}$ choices of random bits, PCP verifier returns NO $\Rightarrow$ NP verifier will return NO.

Real difficulty is proving $NP \subseteq PCP_{1,\frac{1}{2}}(O(\log n), O(1))$.

Why is this useful for proving hardness of approximation?

Intuition: Let Π be arbitrary NP-complete problem (e.g. SAT).
~~Consider problem Π'~~
Then by PCP thm: $\exists$ verifier w/ $O(\log n)$ random bits, $O(1)$ queries, $c(n) = 1$, $s(n) = \frac{1}{2}$ $\Rightarrow$

Let Π' be problem of designing a proof to maximize probability that verifier says YES.
By def, hard to approximate better than $\frac{1}{2}$.
What kind of problem? A CSP!

For each of the poly(n) choices of random bits, verifier computed a (deterministic) function $f(x_1, \ldots, x_k)$, where $x_1, \ldots, x_k$ are the proof bits queried
$\Rightarrow$ poly(n) constraints, each a fn of k bits; max # satisfied constraints
If have soundness $s(n)$, completeness $c(n)$, hard to approx better than $\frac{s(n)}{c(n)}$.

By rewriting arbitrary fns as 3CNF formulae, gives $\frac{15}{16}$-hardness for Max-3SAT.

Can do better by restricting these functions through different versions of PCP thm
<u>Def</u>: Let $odd(x_1, x_2, x_3) = 1$ if $x_1+x_2+x_3$ odd, 0 o/w
$even(x_1, x_2, x_3) = 1$ if $x_1+x_2+x_3$ even, 0 o/w
<u>Thm</u> (Håstad): For any constants $\varepsilon, \delta > 0$,
$NP \subseteq PCP_{1-\varepsilon, \frac{1}{2}+\delta}(O(\log n), 3)$ and verifier restricted to odd and even fns.

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

$\Rightarrow$

Pf: Start w/ arbitrary NP-complete problem, say ~~3~~SAT, and instance φ ~~and instance~~ φ

By Håstad, $\exists$ verifier w/ completeness $1-\varepsilon$, soundness $\frac{1}{2}+\varepsilon$

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

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

Each of N random strings gives three bits and an odd/even test.

We will construct Max-3SAT instance $f(\varphi)$ out of these N tests.

For each ~~odd test~~ $\text{odd}(x_i, x_j, x_k)$ constraint, create 4 clauses:

$x_i \vee x_j \vee x_k$

$\bar{x}_i \vee \bar{x}_j \vee x_k$ $\Rightarrow$ if $\text{odd}(x_i, x_j, x_k)$ satisfied, all 4 ~~constraints~~ clauses satisfied

$\bar{x}_i \vee x_j \vee \bar{x}_k$ if $\text{odd}(x_i, x_j, x_k)$ not satisfied,

$x_i \vee \bar{x}_j \vee \bar{x}_k$ then exactly 3 clauses satisfied

For each $\text{even}(x_i, x_j, x_k)$ constraint, 4 clauses:

$\bar{x}_i \vee x_j \vee x_k$

$x_i \vee \bar{x}_j \vee x_k$ $\Rightarrow$ if $\text{even}(x_i, x_j, x_k)$ satisfied, all 4 clauses satisfied

$x_i \vee x_j \vee \bar{x}_k$ o/w exactly 3 satisfied.

$\bar{x}_i \vee \bar{x}_j \vee \bar{x}_k$

Consider instance φ. If $\varphi \in$ SAT (YES instance), then $\exists$ proof s.t. verifier accepts w.p. $\geq 1-\varepsilon$ (Håstad).

$\Rightarrow$ there is a way of satisfying $\geq 4\cdot(1-\varepsilon)N + 3\varepsilon N$

$= (4-\varepsilon)N$ clauses of $f(\varphi)$

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

Then in any assignment for $f(\varphi)$, can only satisfy all 4 clauses for $\leq (\frac{1}{2}+\varepsilon)N$ constraints

$\Rightarrow$

any assignment for $f(\varphi)$ satisfies $\leq$

$4(\frac{1}{2}+\epsilon)\cdot N + 3(\frac{1}{2}-\epsilon)N = (\frac{7}{2}+\epsilon)N$ clauses

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