Topics (April 7th).

- Hardness Amplification
- Alternate Characterization of NP.
- Probabilistically Checkable Proofs

Hardness Amplification: Example of INDSET.

Theorem: If INDSET is hard to approximate to a factor of 1.001, then it is also hard to approximate to a factor 1000.

Proof: Tensored graph.

General Technique: "Parallel repetition".

NP as a prover-verifier "game":

Classic definition: a language L is said to be in NP if there exists an algorithm A s.t.
1) \( \forall x \in L, \exists w \text{ a witness of poly size such that } A(x, w) = 1 \), and

2) \( \forall x \notin L, \text{ there is no } w \text{ s.t. } A(x, w) = 1 \).

A is a "witness verifier."

E.g. for SAT, A could simply check if \( w \) is a satisfying assignment.

Diagram:

- has \( \emptyset \)
- runs \( A(\emptyset, w) \) and accepts if \( w \) satisfies all clauses.

**PROBABILISTIC CHECKING:** Verifier does not check all clauses. He simply checks one clause at random (lazy verifier)
**KEY:** Prover does not know which clause verifier will check!

Best strategy for prover: provide an assignment that satisfies as many clauses as possible.

Prover solves $\text{max-SAT}$!

Success probability $= \frac{1}{m} \cdot \text{OPT}(\text{max-SAT}(\phi))$

$\Downarrow$

total # of clauses.

**Note:** The lazy prover above:

- Uses $O(\log m)$ "bits of randomness"
- Probes at most 3 bits of proof
- Accepts w.p. $\leq \frac{1}{m}$ if $\phi$ is SAT assuming the witness is correct
  \begin{align*}
  \frac{1}{m} & \quad \text{if } \phi \text{ is not SAT.}
  \end{align*}
**Definition:** An $(r, q)$-verifier is one that uses $r$ bits of randomness, and queries $q$ bits of the proof before accepting/rejecting.

*it could do an arbitrary polynomial amount of computation involving $\phi$ and these bits*

**Definition:** A language $L$ has a PCP $(r, q)$ if there exists an $(r, q)$ verifier $A$ s.t.:

- $\forall x \in L$, there exists a $w$ s.t.
  \[ \Pr[A(x, w) = 1] = 1. \]

- $\forall x \notin L$, there is no $w$ s.t.
  \[ \Pr[A(x, w) = 1] \geq \frac{1}{2}. \]

  (equivalently, $\forall w$, $\Pr[A(x, w)] < \frac{1}{2}$)

**PCP Theorem:** (ALMSS 92).

*Every language $\mathbf{L} \in \mathbf{NP}$ has a PCP$(0(\log n), 0(1))$.*
If a way of writing proofs s.t. even a lazy prover "succeeds" w.p. $\geq \frac{1}{2}$ [Error correction "built-in"]

We now show: PCP Theorem $\Rightarrow$ 3-SAT is hard to approximate to a factor $1+\varepsilon$, for some const $\varepsilon > 0$ (depends on the $O(1)$ term in PCP(..., ...)).

Proof: Let $L$ be some NP-complete language, and consider an $(O(\log n), O(1))$ verifier.

- The algorithm can possibly only query $2^{O(\log n)} = n^{O(1)}$ bits of $w$ [no matter what] bits are.
- It makes its decision based on $q$ of these bits. I.e., based on whether:

$$C_{\phi_k} (w_1, w_2, \ldots, w_q) = 1,$$

for some
We can encode $C_{\phi_R}(...)$ as a SAT formula of size $2^q \Rightarrow 3$-SAT of size $\approx q \cdot 2^q \approx O(1)$.

Thus, since we start with a good verifier, we get that:

- If $x \in L$, "final" 3-SAT formula is satisfiable.
- If $x \notin L$, final 3-SAT formula has at least a $\frac{1}{2} \cdot \frac{1}{2^q}$ fraction of clauses being unsat.

[Completes the proof]