Today

- Multi-prover interactive proofs.
- Oracle interactive proofs.
- Probabilistically checkable proofs.
- NP $\subseteq$ PCP$[O(\log n), \mathrm{poly}(\log n)]$.

Stronger models of proofs?

- Suppose we have *two* provers P_1 and P_2.
- Provers attempting to convince verifier that $w \in L$.
- Can develop common strategy after seeing w.
- But once interaction with verifier starts, they can't communicate with each other.
- Like interrogating two convicts on a common crime!
- Can we prove more this way?

- Introduced by Ben-Or, Goldwasser, Kilian, & Wigderson.
- Motivation: Get "zero-knowledge" proofs without cryptographic assumptions.

Multi-prover interactive proofs (MIP)

$L \in 2\mathrm{IP}$ if there exists a polynomial time bounded verifier V interacting with *two* provers satisfying the following properties:

Completeness $w \in L$ implies there exist P_1, P_2 such that $\Pr[P_1 \leftrightarrow V \leftrightarrow P_2 \text{ accepts}] = 1$.

Soundness $w \notin L$ implies for every P_1, P_2 $\Pr[P_1 \leftrightarrow V \leftrightarrow P_2 \text{ accepts}] \leq 1/3$.

Multi(!)-prover interactive proofs (MIP)

- Above definition restricts to *two* provers.
- Robust w.r.t. error, one-sided vs. two-sided etc.
- What about more provers? three? four? poly?
- Can extend definition easily. Power?

Oracle interactive proofs (OIP)

- Prover fixes a function $f : \mathcal{Q} \to \mathcal{A}$ ($\mathcal{Q}$ is question space, i.e., $\{0,1\}^{\text{poly}}$; and $\mathcal{A}$ is answer space).
- Verifier interacts with the "oracle" for function f.
- Model introduced by Fortnow, Rompel & Sipser.

OIP vs. MIP

- Oracle can simulate any number of provers! (Questions to prover P_i can be simulated by a query of the form $(i, \mathbf{h})$, where $\mathbf{h}$ is the entire history of questions to P_i so far.)
- Proposition [FRS]: Oracle can be simulated by two provers.
- Proof idea: If verifier is non-adaptive, then the following simulates the conversation. Say verifier wishes to query f for $q_1, \ldots, q_m$. Send $\langle q_1, \ldots, q_m \rangle$ to P_1 and q_j (for random j) to P_2. P_1 expects to respond with $f(q_1), \ldots, f(q_m)$ and P_2 with $f(q_j)$. Say they respond with $a_1, \ldots, a_m$

and b. MIP V accepts if OIP verifier accepts $a_1, \ldots, a_m$ and $a_j = b$.

- Completeness, soundness = exercise.
- Adaptive verifier case = exercise.

Power of MIP, OIP = ?

- We know $2\mathrm{IP}$ has same power as OIP. But is this more than IP?

Theorem [Babai, Fortnow, Lund]: MIP = NEXPTIME.

So, given our current state of knowledge, MIP seems more powerful.

Will see some version of theorem in the next few lectures.

Digesting MIP=NEXPTIME

- NEXPTIME is just proving theorems, where the proofs are exponentially long in the theorem. (So if we pad the theorem, this just looks like NP.)

- MIP = OIP. What does OIP look like? The oracle is just another big proof, also exponential sized in the theorem. Only now the verifier is probabilistic; runs in polynomial time; and errs when $w \notin L$.

- Can simulate verifier on all random strings and the new one runs in exponential time. So OIP is really just a restriction of NEXPTIME; but BFL theorem says it is equally powerful.

Scaling MIP down to NP

- Does MIP=NEXP phenomenon have analog for NP?

- Not if we track verifier's running time. It is polynomial for NP, and needs to be linear to do anything interesting.

- But other features interesting.

- Randomness is small in proof size.

- Number of queries to proof is small (poly logarithmic in proof size).

- No reason why this aspect can not scale down to NP.

Probabilistically checkable proofs (PCPs)

- PCP verifier = OIP verifier.
  - Runs in prob. poly time.
  - Tosses coins.
  - Makes few queries.

- Quantifying resources: (r, q)-restricted PCP verifier is an OIP verifier that tosses $r(n)$ coins and queries proof oracle at most $q(n)$ times.

- $\mathrm{PCP}_{c,s}[r, q]$: Class of all languages with (r, q) restricted PCP verifier, with completeness c and soundness s.

Optimal prover & Hardness of Max SAT

Show that determining optimal prover for a given PCP reduces to a satisfiability problem.

Since approximating acceptance probability of optimal prover suffices to distinguish complete cases from sound cases, it follows that approximating MAX SAT is NP-hard.