Notes for Lecture 20
Malka Rosenthal

Definition 1 Probalistically Checkable Proofs,$PCP(r(n), q(n))$ *is defined (for any functions r and q) as the class of languages, L, that are accepted by polynomial size machines as follows: On input* $x, |x| = n$, *the machine uses* $O(r(n))$ *random bits and queries the proof in only* $O(q(n))$ *places. If* $x \in L$, *the machine will accept with probability 1 and if* $x \notin L$, *the machine rejects with probability* $\geq 3/4$.

On every input, the machine is given both the input and a proof.

Theorem 1 $\mathcal{NP} \subseteq PCP(\log n, \log^{O(1)} n)$

The following claims are easily seen to be true:

Claim 1 $NP = PCP(0, n^{O(1)})$

That $NP \subseteq PCP(0, n^{O(1)})$ is obvious: simply let the proof be an accepting branch of the nondeterministic Turing machine which accepts L.
To see that the opposite direction ($PCP \subseteq NP$) is equally obvious, just note that there is no point in having a proof longer than polynomial if there is no randomness and the machine is restricted to run in polynomial time.

Claim 2 $PCP(\log n, \log n) \subseteq NP$

Proof: Consider all possible random sequences of length $O(\log n)$. For each sequence, look at which bits will be queried; being that each query depends on the previous queries, there will actually be a binary tree of size $2^{O(\log n)} = n^{O(1)}$ representing the $O(\log n)$ queries for the given sequence. In all, there will be $n^{O(1)}$ query bits for each of the $n^{O(1)}$ possible random sequences. Nondeterministically, guess these polynomially many bits (i.e. you will be guessing the proof).
Accept the input if all guesses cause to accept and reject if at least one causes to reject.

Note It will suffice to show that 3-SAT has a proof system of this form.(3-SAT is the set of formulae in CNF such that each clause has at most 3 literals which are satisfiable.)

This is because 3-SAT is NP-Complete:

$$\forall L \in \mathcal{NP}, \exists f \text{ s.t. } x \in L \Leftrightarrow f(x) \in \text{ 3-SAT.}$$

Thus, given a proof system for 3-SAT, the proof y for x will be the proof that $f(x) \in$ 3-SAT.

Proof of Theorem 1 In order to prove the theorem, we will construct an arithmetic version of 3-SAT as follows:
Consider φ, a formula in 3-CNF. Let the clauses be $C_1, \ldots, C_m$ and the variables $v_1, \ldots, v_m$. (If necessary, fill in some dummy variables or clauses to make the same number of each.) Define functions $\chi_1, \chi_2, \chi_3, S_1, S_2, S_3$ as follows:

- For $1 \leq l \leq 3$,

$$\chi_l(i,j) = \begin{cases} 1 & \text{if the } l^{th} \text{variable in} C_i \text{ is } v_j \\ 0 & \text{otherwise} \end{cases}$$

- For $1 \leq l \leq 3$,

$$S_l(i) = \begin{cases} 1 & l^{th} \text{ variable in } S_i \text{ is } \underline{\text{not}} \text{ negated} \\ 0 & \text{otherwise} \end{cases}$$

Note Clearly, $\chi_1, \chi_2, \chi_3, S_1, S_2, S_3$ completely describe φ and are computable given φ.
If $A(v)$ is an assignment to the variable v, then clause C_i is satisfied by A iff

$$\sum_{v_1, v_2, v_3} \prod_{j=1}^{3} \chi_j(i,j)(S_j(i) \Leftrightarrow A(v_j)) = 0.$$

(Further lectures on PCP were based on the text by Arora, available by ftp from ECCC.)