In this lecture, by a PRG we mean the definition where the generator is assumed secure with respect to tests running in specific time. This should be contrasted with the BMY-style definition where we assume that the generator is secure against all (non/uniform) polynomial time tests.

**Theorem 1** \( \forall t(n), \exists \text{PRG}, G : \{0,1\}^{\log t(n)} \rightarrow \{0,1\}^t(n) \text{ that is secure against nonuniform } t(n). \)

**Proof:** Take \( G \) at random. Then for any fixed \( t(n) \)-time \( T \), Chernoff inequality implies, 
\[
\Pr_{\text{choose } G} \left[ T \text{ breaks } G \right] \leq 2^{\Omega(2^{O(\log t)})}
\]

The number of all possible nonuniform time \( t(n) \) tests is at most \( 2^{t^2(n)} \). So, 
\[
\Pr[\exists t(n) \text{ time test, breaking } G] \leq 2^{t^2(n)}2^{\Omega(2^{O(\log t)})} \ll 1
\]
for \( t = \frac{1}{\epsilon} \) and taking large constant in \( O(\log t) \).

**Corollary 2** \( \text{BPP} \subseteq \text{non-uniformP} \).

(For any \( \text{BPP} \) algorithm, there is a family of polynomial size circuits that decide the same language.)

An other notation for non-uniform\( \text{P} \) is \( \text{P/poly} \).

**Proof:** (idea) Non-uniformity gives us a way to "hardwire" a PRG, \( G : \{0,1\}^{O(\log t)} \rightarrow \{0,1\}^t \) for each input length. Then we can use this "hardwired" PRG to simulate true randomness of a given \( \text{BPP} \)-algorithm.

**Big Open:** How to construct such PRG’s efficiently, uniformly?

**Theorem 3** If there is a BMY-style PRG, then \( \text{P} \neq \text{NP} \).

**Proof:** Contrapositive \( \text{P} = \text{NP} \implies \exists \text{BMY-style PRG} \)

This algorithm breaks any PRG, \( G : \{0,1\}^{t(n)} \rightarrow \{0,1\}^{t(n)} \):

- Given \( x \in \{0,1\}^n \)
- non-deterministically guess \( s \in \{0,1\}^{t(n)} \)
- if \( G(s) = x \), then Accept else Reject.

Since \( \text{P} = \text{NP} \) the above algorithm can be done in polynomial time.
For **BPP** derandomization it is sufficient to have a PRG secure against specific poly-time rather than all possible poly-tests. In particular, PRG may run in more time than the test.

**Remark 4** $P = NP$ implies there is a PRG secure against fixed poly-time.

**Remark 5** Existence of PRG secure against fixed poly-time implies lower bounds proofs.

**Definition 6 (one-way function (OWF))** A function $f : \{0,1\}^* \rightarrow \{0,1\}^*$ is a one-way function if:

- (1) $f$ is polynomial time (probabilistic) computable.
- (2) For any probabilistic polynomial time algorithm $A$, $\Pr[A(1^n, f(x)) \in f^{-1}(f(x))]$ is less than any inverse polynomial.

**Theorem 7 (Høastad, Impagliazzo, Levin, Luby)** \( \exists \text{OWF} \iff \exists \text{BMY-style PRG} \).