Class of Randomized Algorithms and Derandomization.

**Deterministic Algorithms.** (For simplicity, we focus on decision problems now.)

\[ \xrightarrow{\text{input bits}} A \xrightarrow{\text{output}} 0/1 \ (\text{YES/NO}). \]

(ideally we want \( A \) to run in polynomial time, say time \( n^k \))

**Randomized Algorithms.** \( A \) is still deterministic, but has “random bits”

\[ \xrightarrow{\text{actual input}} A \xrightarrow{\text{output}} 0/1. \]

\( r \) “random bits”

Example. Miller–Rabin Primality testing: \( x \) is the \( n \) to be tested.

\( A \) solves a problem in “BPP” if \( \forall x \)

\[ \Pr_r [A(x,r) \text{ correct}] \geq \frac{3}{4} \]

Remark 1 “\( \frac{3}{4} \)” can be any \( c \in (\frac{2}{3}, 1) \). To make it \( 1 - \epsilon \), we run \( A(x,r) \) \( O(\log n) \) times indep. and take the majority vote.

Remark 2 If \( A \) only requires \( O(\log n) \) random bits, it’s trivial to make \( A \) deterministic. — Simply try all \( 2^{O(\log n)} = \text{poly}(n) \) possible \( r \)’s, and take the majority vote.

Derandomization. (Is BPP = P?) How to make \( A \) deterministic even if it uses \( \omega(\log n) \) random bits?

**Pseudorandom Generator.** (PRG) Let \( C \) be a class of fn’s \( f : \{0,1\}^n \rightarrow \{0,1\} \). \( G : \{0,1\}^* \rightarrow \{0,1\}^n \)

\( l < n \) is an \( \epsilon \)-PRG for \( C \) if with seed length \( l \) if

\[ \forall f \in C : \left| \Pr_{s \sim \{0,1\}^l} [f(G(s)) = 1] - \Pr_{r \sim \{0,1\}^n} [f(r) = 1] \right| < \epsilon. \quad \text{“} G \ v-fools \ C \text{”} \]

Typically, want \( G(s) \) computable in \( \text{poly}(n) \) time (deterministically).

Intuition \( C \) not able to distinguish between distrib. \( \{G(s)\}_{s \sim \{0,1\}^l} \) and uniform distr. \( \{0,1\}^n \)

However \( \{G(s)\} \) has a much smaller support.

Example. Say \( A \) runs in \( n^k \) time, uses \( n \) random bits. Let \( C \) be \( \{f : \{0,1\}^n \rightarrow \{0,1\}, f \text{ computable in } n^k \text{ time}\} \). If \( G \ v-fools \ C \), then

\[ \Pr_{s \sim \{0,1\}^n} [A(G(s)) \text{ correct}] \geq \Pr_{r \sim \{0,1\}^n} [A(r) \text{ correct}] - 0.1 \geq \frac{3}{4} - 0.1 = 0.65 \]

A deterministic alg. to enumerate \( s \) and take maj. vote: runs in \( 2^{\text{poly}(n)} \) time.
If \( l = \omega(\log n) \), the algorithm solves \( A \) in \( P \).

**Theorem [Impagliazzo-Widgerson '97]** Suppose \( \forall m \exists m : \{0,1 \}^m \rightarrow \{0,1 \}^m \) computable in time \( 2^{\omega(m)} \) but not in time \( 2^{o(m)} \), then there is a PRG \( \epsilon \)-fools all poly-time algorithms with seed length \( \omega(\log n) \), i.e. \( \text{BPP} = \text{P} \). (The assumption is stronger than \( P \neq \text{NP} \) but believable.)

**Intuition.** A function hard to compute \( \Rightarrow \) looks random to Turing Machines with less time resource \( \Rightarrow \) fools these TMs.

**k-wise Independent PRGs.** \( G : \{0,1 \}^l \rightarrow \{0,1 \}^n \) is \( k \)-wise indep. if:
- \( \forall s \in \{0,1 \}^n \quad \Pr \left[ (G(s))_i = 1 \right] = \frac{1}{2} \)
- \( \forall 1 \leq i_1 < i_2 < \ldots < i_k \leq n \) the distribution \( (G(s))_{i_1}, (G(s))_{i_2}, \ldots, (G(s))_{i_k} \) is uniform on \( \{0,1 \}^k \)

**Constructing pairwise indep. PRGs.** \( G : \{0,1 \}^l \rightarrow \{0,1 \}^{2^{l-1}} \) defined as

\[
G(s) = s \mod 2 \quad \text{for all } s \in \{0,1 \}^l, \quad v \mod 5
\]

**Proof.** \( \forall \nu \neq 0 : \quad \Pr \left[ s \mod 2 = 1 \right] = \frac{1}{2} \)
\( \forall v_1 \neq v_2 : \Pr \left[ s \mod 2 \neq (s, v) \mod 2 \right] = \Pr \left[ (s,v) \mod 2 = 0 \right] = \frac{1}{2} \)

Recall Hadamard Code.

**Theorem [Alon-Babai-Itai '85]** \( \forall k \leq n, \text{ prime power } q, \exists \text{ poly-time computable } k \)-wise indep. generator with \( l = \left\lfloor \frac{k}{2} \right\rfloor \log n + O(1) \).

**Application.** Derandomize the following algorithm for Max-Cut.

**Max-Cut.** Given \( G = (V, E) \), find \( S \subseteq V \) to maximize \( |\text{edges}(S, V \setminus S)| \)

**Alg.** For each \( i \in V \), toss \( G \in \{0,1 \} \), \( i \in S \) iff \( G_i = 1 \)

**Analysis.** \( E \left| \text{edges}(S, V \setminus S) \right| = \sum_{v \in V} \sum_{i \in S} E[l_i + \bar{f}_i] \)

\[
\leq \sum_{v \in V} \sum_{i \in S} \Pr [l_i + \bar{f}_i] = \sum_{v \in V} E \frac{|E|}{2} = \frac{|E|}{2} \left\Rightarrow \begin{array}{c}
\text{cut at least 50% edges} \\
\text{not bad.}
\end{array} \right.
\]

\[
\uparrow \quad \text{linearity of expectation pairwise indep.}
\]

**Observation.** \( r \in \{0,1 \}^n \) be pairwise indep. suffices for the analysis.

**Use** \( r - G(s) \) where \( s \in \{0,1 \}^n \), \( G \) pairwise indep.

**Enumerate** \( S \) in polynomial-time.
\( \varepsilon \)-Biased Generators: \( G : \mathbb{F}_2^l \rightarrow \mathbb{F}_2^n \) is an \( \varepsilon \)-biased generator if
\[
\forall w \in \mathbb{F}_2^n, w \neq 0, \quad \Pr_{s \sim \mathbb{F}_2^l}[w \cdot G(s) = 1] \in \left[ \frac{1}{2} - \frac{\varepsilon}{2}, \frac{1}{2} + \frac{\varepsilon}{2} \right]
\]

It fools all degree-1/linear functions.

**Theorem [NN'93]** \( l = O(\log n \log \frac{1}{\varepsilon}) \) achievable w/ \( G \) poly-time computable.

**[AGHP'92]** \( l = 2\log \frac{n}{\varepsilon} + o(1), \ O\left(\frac{1}{\varepsilon}\right)\)-time computable.

**Application.**

**Input:** \( A, B, C \in \mathbb{F}_2^{\text{num}} \).

**Goal:** Check \( AB = C \) in \( O(\varepsilon^2) \) time.

**Alg:** Choose \( y \sim \mathbb{F}_2^n \) uniformly, check if
\[
\frac{(AB)y = Cy}{\text{(accept)}} \quad \text{in } O(\varepsilon^2) \text{ time}
\]

\[
\frac{A(By)}{\text{(rejected)}} \quad \text{in } O(\varepsilon) \text{ time}
\]

**Analysis:**

When \( AB = C \) \( \Rightarrow \ \Pr[(AB)y = Cy] = 1 \)

When \( AB \neq C \) \( \Rightarrow D = AB - C \) has \( \geq 1 \) non-zero row, namely of
\[
\Pr[(AB)y = Cy] = \Pr[Dy = 0] \leq \Pr[d \cdot y = 0] = \frac{1}{2}
\]

[Can repeat w/ several \( y \) to gain high confidence]

Uses \( O(\varepsilon^2) \) time, \( n \) random bits.

If \( y \) is output of a \( \varepsilon \)-biased gen. \( \Pr[d \cdot y = 0] \leq \frac{1}{2} + \frac{1}{2} = .55 \)

\( \Rightarrow O(\varepsilon^2) \) time, \( O(\log n) \) random bits. (using [AGHP'92])