Pseudorandom Generator

- Functionality
  - Deterministic algorithm G
  - Takes as input a short random seed s
  - Ouputs a long string $G(s)$
- Security
  - No efficient algorithm can "distinguish" $G(s)$ from a truly random string r.
  - i.e. passes all "statistical tests."
- Intuition:
  - Stretches a small amount of true randomness to a larger amount of pseudorandomness.
- Why is this useful?
  - We will see that pseudorandom generators will allow us to beat the Shannon bound of $|\boldsymbol{K}| \geq |\boldsymbol{M}|$.
  - I.e. we will build a computationally secure encryption scheme with $|\boldsymbol{K}| < |\boldsymbol{M}|$

Pseudorandom Generators

Definition: Let $\ell(\cdot)$ be a polynomial and let G be a deterministic poly-time algorithm such that for any input $s \in \{0,1\}^n$, algorithm G outputs a string of length $\ell(n)$. We say that G is a pseudorandom generator if the following two conditions hold:

1. (Expansion:) For every n it holds that $\ell(n) > n$.
2. (Pseudorandomness:) For all ppt distinguishers D, there exists a negligible function $negl$ such that:

$$\left|\Pr[D(r) = 1] - \Pr\left[D\big(G(s)\big) = 1\right]\right| \leq negl(n),$$

where r is chosen uniformly at random from $\{0,1\}^{\ell(n)}$, the seed s is chosen uniformly at random from $\{0,1\}^n$, and the probabilities are taken over the random coins used by D and the choice of r and s.

The function $\ell(\cdot)$ is called the expansion factor of G.