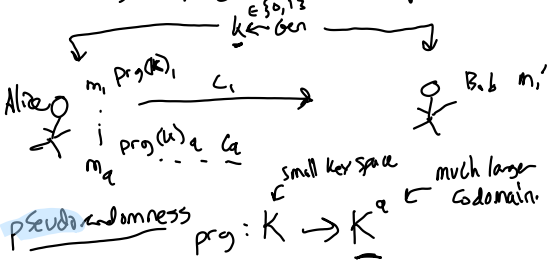
Pseudorandomness

Monday, August 30, 2021 3:58 PM

Founding Cryptography on Computational assumptions



pseudorandomness $prg: K \to K^q$

"considered to be random, as long as you don't know the seed"

$PRG: K \to C$

"without the seed, output of PRG indistinguishable from truly random samples"

$$\forall A. \left| \Pr\left[\begin{array}{l} k \xleftarrow{\$} K \\ x \leftarrow prg(k) \\ b \leftarrow A(x) \end{array}\right]_{b=0} - \Pr\left[\begin{array}{l} x \xleftarrow{\$} C \\ b \leftarrow A(x) \end{array}\right]_{b=0} \right| \leq negl(\lambda)$$

(real: $k \xleftarrow{\$} K$, $x \leftarrow prg(k)$; random: $x \xleftarrow{\$} C$; A: any adversary; any PPT)

Problems thought to be hard

- prime factorization

think ∄ an algorithm $prog(n)$ that outputs prime factorization of input n and takes #steps $\leq p(|n|)$ for some polynomial p.

- Satisfiability for circuits, circuit w/ #n inputs
- hamiltonian - knapsack - tsp.
- distinguishing PRGs
- one way functions

given $f: D \to C$

$$\forall A. \Pr\left[\begin{array}{l} x \xleftarrow{\$} D \\ y \leftarrow f(x) \\ x' \leftarrow A(y) \\ \hline f(x') = y \end{array}\right] \leq negl(\lambda)$$

PPT: polynomial time probabilistic TM.

} security

$M(1^\lambda, \ldots; z)$ terminates in $poly(\lambda)$ steps.

- Negligible

"function that gets small faster than any polynomial $\left(\frac{1}{p(\lambda)} \text{ for any poly } p(\lambda)\right)$"

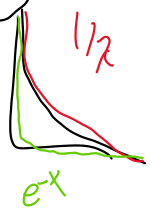
- $f(\lambda)$ is $\text{negl}(\lambda)$ if

$\forall$ poly $p(\lambda)$, $\lim_{\lambda \to \infty} p(\lambda) f(\lambda) = 0$

ex. $f(\lambda) = 0$ ✓ $f(\lambda) = 1$ ✗ not negligible

$f(\lambda) = \frac{1}{\lambda^2}$ w, $p(\lambda) = \lambda^3$ $p(\lambda) \cdot f(\lambda) = \lambda$ ✓

$f(\lambda) = \frac{1}{2^{\sqrt{\lambda}}}$ for next time $f(\lambda) = e^{-\lambda}$



"let c be the degree of $p(\lambda)$"

$\forall p(\lambda)$ let c be the degree of $p(\lambda)$

Pf. $\lim_{\lambda \to \infty} p(\lambda) \frac{1}{2^{\sqrt{\lambda}}}$ λ^c

goal: $\forall p(\lambda)$

simplify

$\forall c$, $\lim_{\lambda \to \infty} \frac{\lambda^c}{2^{\lambda^{1/2}}} = 2^{c \log_2 \lambda} \cdot 2^{-\lambda^{1/2}} = 2^{c \log_2 \lambda - \lambda^{1/2}}$

$\lim_{\lambda \to \infty} c \log \lambda - \lambda^{1/2} = -\infty$ ✓

A_1 (prob) → solution, succeeds with $p(\lambda) = \text{negl}(\lambda)$ success.

A_2 (prob):

run A_1 (prob) $q(\lambda)$ times for some poly $q(\lambda)$.

A_2 success $\leq q(\lambda) \cdot p(\lambda)$

This is still negligible.

by Union Bd. $\Pr[\text{succ}] \leq \sum_{i}^{q(\lambda)} p(\lambda)$

Union Bound: $\Pr[A_1 \cup A_2 \cdots \cup A_k] \leq \sum \Pr[A_i]$.

ex. Roll a 6-

give UB for $\Pr[X \text{ is even} \cup X \text{ is } 4 \cup X < 5]$

Proof: $= 1/2 + 1/6 + 5/6 = 9/6.$

$$\Pr[A \cup B] = \Pr[A] + \Pr[B] - \Pr[A \cap B]$$

Birthday Problem

Distinguish sampling w/o replacement vs w/ replacement.

w/ replacement → $O_1()$:
$r \xleftarrow{\$} \{0,1\}^n$
return r

w/o replacement. $O_2()$:
initially $R := \emptyset$
$r \xleftarrow{\$} \{0,1\}^n \setminus R$
$R := R \cup \{r\}$
return r

Indistinguishable: "A runs and has oracle access to O"

$$\forall A. \quad \{A^{O_1}\} \approx \{A^{O_2}\}$$

$$\left| \Pr\left[b \leftarrow A^{O_1}(1^n) : b = 0 \right] - \Pr\left[b \leftarrow A^{O_2}(1^n) : b = 0 \right] \right| \leq \text{negl}(n).$$

Proof: ??

Focus on an attacker A^O that does...

$X_1 \leftarrow O()$
$\vdots$
$X_{q(n)} \leftarrow O()$

Some poly $q(n)$

Any duplicates? If so output 0 else 1..

O_3: $R := \emptyset$ initially

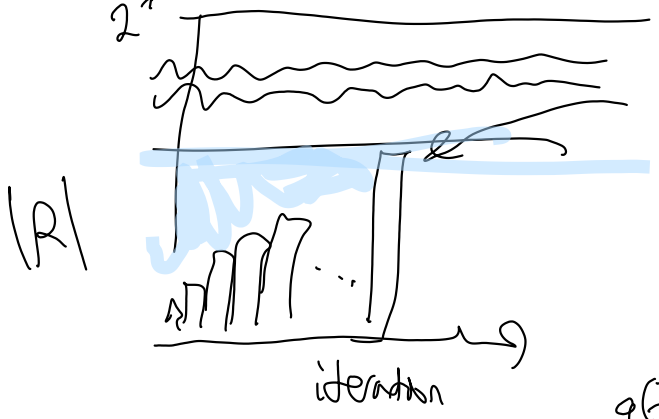
$r \in \{0,1\}^\lambda$

if $r \in R$: BadEvent := 1

$R := R \cup \{r\}$

return r

What's Pr of BadEvent=1 in O_3 for $\mathcal{A}$?



Let $q(z)$ be a polynomial

on ith iteration

$|R| = q(i)$

Pr [bad event on step i given not bad ...]

$$\Pr[\text{BadEvent occurs at all}] \leq \sum_{i=0}^{q(z)} \Pr[\text{BadEvent occurs at } i \text{ but not earlier}]$$

$$\sum_{i=0}^{q(z)} \frac{i}{2^\lambda} \leq \sum^{q(z)} \max(\text{each } \in$$

$$\leq \sum \frac{q(z)}{2^\lambda}$$

$$\leq \frac{(q(z))^2}{2^\lambda}$$

$O_1 \overset{s}{\approx} O_3 \approx O_2$

by Bad event lemma.