Notes from the
Student Wish Course on

Zero-Knowledge

Michael Nüsken

b-it

(Bonn-Aachen International Center
for Information Technology)

summer 2007

6.8.07 (1)

(A1) (i) $X = U_{100}$, $Y = g(U_{10})$, g not necessarily injective.

Estimate the statistical difference.

(ii) Consider $X = U_{\ell(n)}$ versus $g(U_n)$.

(A2) (i) Consider $g: \{0,1\}^3 \to \{0,1\}^6$

x	$g(x)$
000	001100
001	001110
010	010101
011	011011
100	101000
101	100101
110	110010
111	110011

Compute the statistical difference between $g(U_3)$ and U_6.

(ii) Consider the distinguisher $D(y)$ that outputs 1 if at most four bits in y are one. ...

(A3) Understand the details of the proof that statistical closeness implies computational indistinguishability.

Proof as a tool:

$$\frac{1}{2} \sum_{\alpha} | \mathrm{prob}(X=\alpha) - \mathrm{prob}(Y=\alpha) |$$

$$= \max_S | \mathrm{prob}(X \in S) - \mathrm{prob}(Y \in S) |$$

(A4) (i) Give a problem / a language is in $\mathcal{NP}$ but not $\mathcal{NP}$-complete.

(ii) ~~Give an~~ language in $\mathcal{NP} \setminus \mathcal{P}$.

(iii) language in $\mathcal{BPP} \setminus \mathcal{P}$.

A 1/2 $\qquad$ 6.8.07 ②

$$\Delta = \frac{1}{2}\sum_{\alpha} \left| \operatorname{prob}(U_{\ell(w)} = \alpha) - \operatorname{prob}(g(U_n) = \alpha) \right|$$

$$= \frac{1}{2}\Big(\sum_{\alpha \in \operatorname{img}} \left(\# g^{-1}(\alpha) \cdot 2^{-n} - 2^{-\ell(w)} \right)$$

$$+ \sum_{\alpha \notin \operatorname{img}} \left(2^{-\ell(w)} - 0 \right) \Big)$$

$$= \frac{1}{2}\left(1 + 2^{-\ell(w)} \left(-\#\operatorname{img} + 2^{\ell(w)} - \#\operatorname{img} \right) \right)$$

$$= \frac{1}{2}\left(2 + \cancel{2^{-\ell(w)}} - 2\#\operatorname{img}\, 2^{-\ell(w)} \right)$$

$$= 1 - 2^{-\ell(w)} \#\operatorname{img}.$$

if $\ell(w) > n$ then $\Delta \geq \frac{1}{2}$.

$(\#\operatorname{img} \leq 2^{n}.)$

A3

Tool

$$\frac{1}{2}\sum_{\alpha} \Big| \underbrace{\operatorname{prob}(X=\alpha) - \operatorname{prob}(Y=\alpha)}_{} \Big|$$

using $S = \{ \alpha \mid \% > 0 \}$.

$$= \frac{1}{2}\sum_{\alpha \in S} \left(\operatorname{prob}(X=\alpha) - \operatorname{prob}(Y=\alpha) \right)$$

$$- \frac{1}{2}\sum_{\alpha \notin S} \left(\operatorname{prob}(X=\alpha) - \operatorname{prob}(Y=\alpha) \right)$$

$$= \frac{1}{2}\left(\underline{\operatorname{prob}(X \in S)} - \operatorname{prob}(Y \in S) \right)$$

$$- \frac{1}{2}\Big(\underbrace{\operatorname{prob}(X \notin S)}_{1-\operatorname{prob}(X \in S)} - \underbrace{\operatorname{prob}(Y \notin S)}_{1-\operatorname{prob}(Y \in S)} \Big)$$

$$= \operatorname{prob}(X \in S) - \operatorname{prob}(Y \in S)$$

$$\geq \max_S \left| \operatorname{prob}(X \in S) - \operatorname{prob}(Y \in S) \right|$$

st. d. $\Rightarrow$ c. i.

Assume that D is a distinguisher, say given as a poly-size circuit C_n, or say it doesn't use randomness.

Then let $S = \{\alpha \mid D(\alpha) = 1\}$.

$$|\mathrm{prob}(\underbrace{D(X)=1}_{X\in S}) - \mathrm{prob}(\underbrace{D(Y)=1}_{Y\in S})|$$

$$= |\mathrm{prob}(X\in S) - \mathrm{prob}(Y\in S)|$$

$$\leq \Delta(X,Y).$$

A2 (i) $\Delta(g(U_3), U_6) = 1 - 2^{-3}$

(ii) $\delta = 1 - \underbrace{\frac{57}{64}}_{=\frac{7}{64}} \leq \frac{7}{8}$.

A4 If $NP \neq P$: (i) any $L \in P$
(ii) any NP-complete.
(iii) BPP

Possibly Graph Isomorphism.

Monday summary

- Randomness?
- ensemble
- computational indistinguishable
- statistically close Lemma: st.cl. => c.i.
- c.i. by repeated sampling
- Thm c.i. <=> c.i.$^{poly(n)}$.

Proof: hybrid technique.

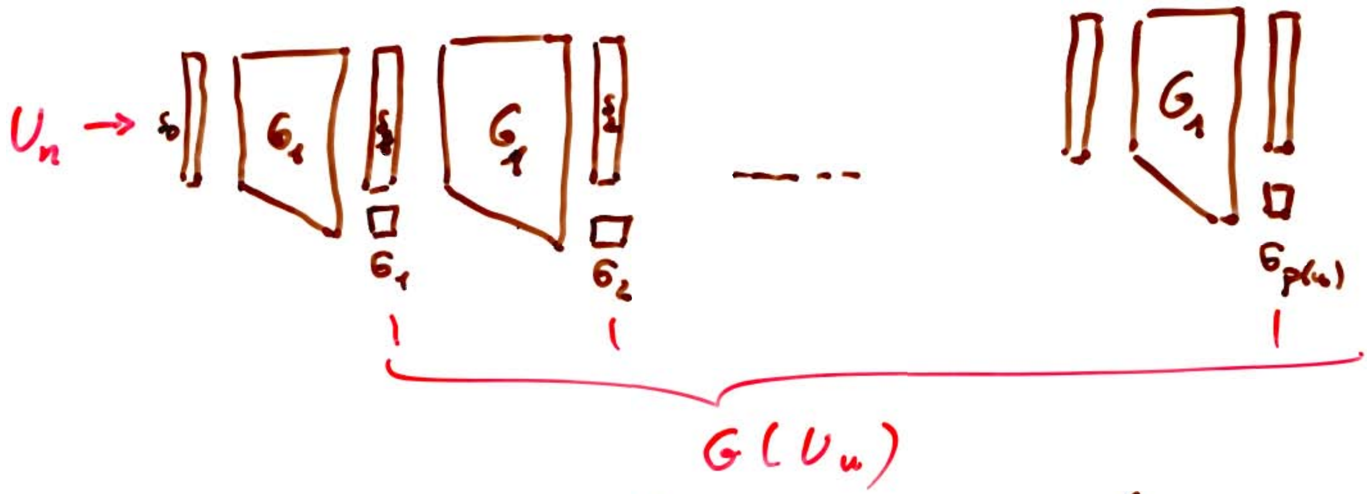
7.8.07 ②

Proof (Then short becomes long)

Construction of the long generator G

Inputs s_0 n-bit string.

Outputs $\sigma_1 \dots \sigma_{p(n)}$ $p(n)$-bit string.



Here, G_1 is a pseudo-random generator with expansion $\ell(n) = n+1$,

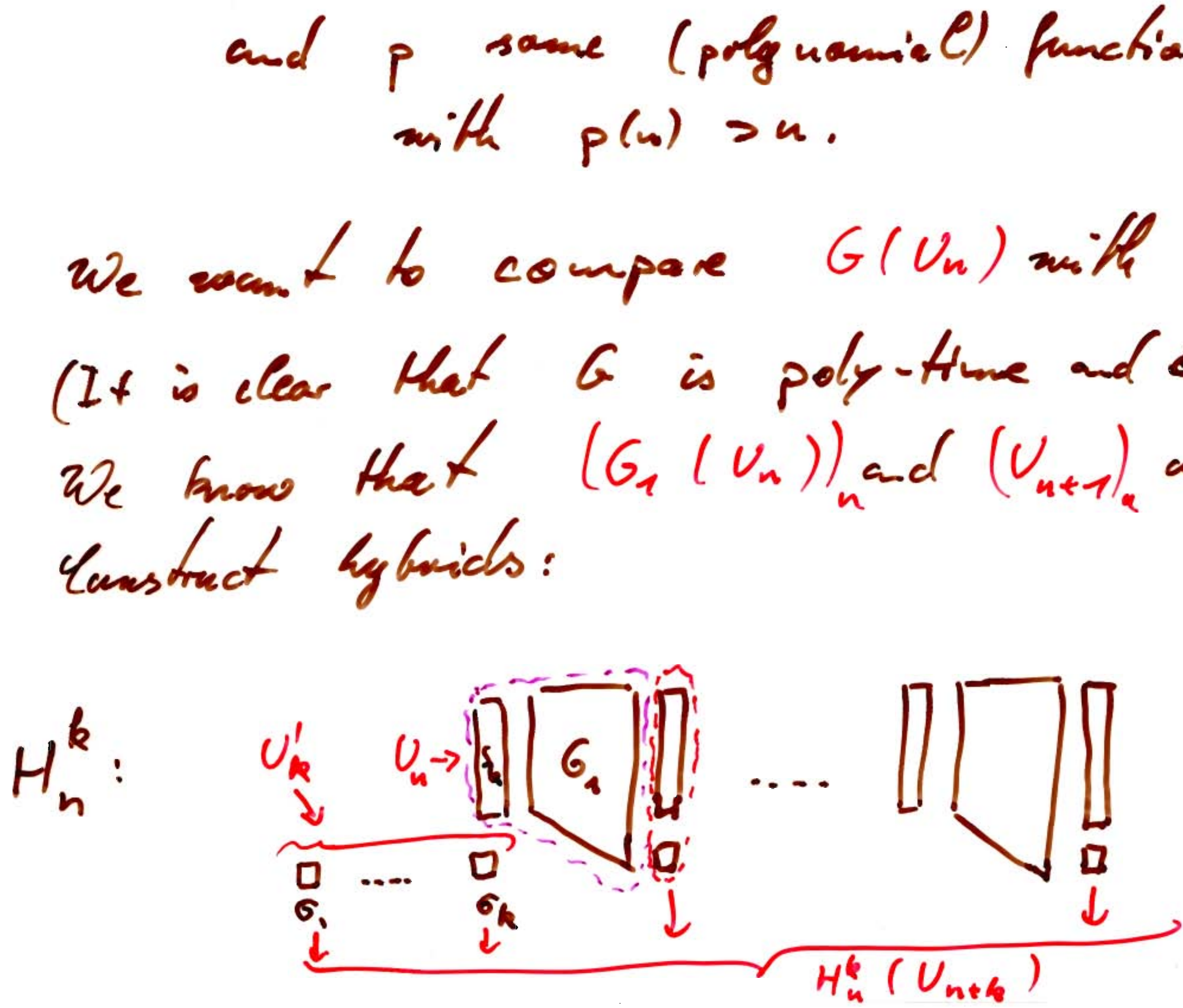
and p some (polynomial) function with $p(n) > n$.

We want to compare $G(U_n)$ with $U_{p(n)}$.

(It is clear that G is poly-time and expanding).

We know that $(G_1(U_n))_n$ and $(U_{n+1})_n$ are c.i.

Construct hybrids:



7.8.07
3

Show:

- Extremes: $H_n^0 = G(U_n)$

  $H_n^{p(n)} = U_{p(n)}$

- Neighbors: H_n^{k+1} : $\sigma_{k+1} s_{k+1} \leftarrow U_{n+1}$

  $\wr_k$

  H_n^k : $\sigma_{k+1} s_{k+1} \leftarrow G_1(U_n)$

⌈ Suppose D_0 tries to distinguish H_n^k and H_n^{k+1}.

Then D_1:

Input: $\sigma_{k+1} s_{k+1}$. $\leftarrow$ $G_1(U_n)$, U_{n+1}

Output: 0 or 1.

1. Choose $\sigma_1 \dots \sigma_k$ at random uniformly.
2. Compute $\sigma_{k+2} \dots \sigma_{p(n)}$ as in the generator starting with seed s_{k+1}.
3. Output $D_0(\sigma_1 \dots \sigma_k \, \sigma_{k+1} \, \sigma_{k+2} \dots \sigma_{p(n)})$

Claim: if D_0 distinguishes H_n^k and H_n^{k+1} then D_1 distinguishes $G_1(U_n)$ and U_{n+1}.

Warning: This is only heuristic.

$$\operatorname{prob}(D_1(G_1(U_n))=1) - \operatorname{prob}(D_1(U_{n+1})=1)$$
$$= \operatorname{prob}(D_0(H_n^k)=1) - \operatorname{prob}(D_0(H_n^{k+1})=1)$$

Thus: ~~if D_0~~ the claim is true. ⌋

- # Hybrids = $p(n)$ polynomial ✓

Suppose D tries to distinguish
$G(U_n)$ and $U_{p(n)}$.

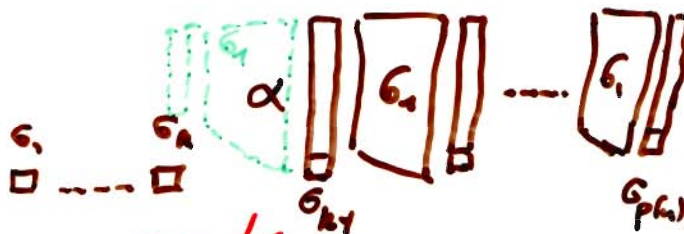
Define D'

Input: $\alpha \in \{0,1\}^{n+1}$

Output: 0 or 1.

~~1. Choose $\sigma_1, \dots,$~~

1. Choose $k \in_R \{0, \dots, p(n)-1\}$ uniformly random.
2. Choose $\sigma_1, \dots, \sigma_k \in_R \{0,1\}$ uniformly random and independently.
3. write $\alpha = G_{k+1}\, \sigma_{k+1}$.
4. Compute $G_{k+2}, \dots, G_{p(n)}$ as in the long generator.



5. Output $D(\underbrace{\sigma_1 \dots \sigma_k}_{U_k} \underbrace{\sigma_{k+1}\, \sigma_{k+2} \cdots \sigma_{p(n)}}_{\text{generated}})$

$$\mathrm{prob}(D'(G_1(U_n)) = 1) = \frac{1}{p(n)} \sum_{k_0=0}^{p(n)-1} \mathrm{prob}(D(H_n^{k_0}) = 1 \cancel{\mid k=k_0})$$

$$= \sum_{k_0=0}^{p(n)-1} \underbrace{\mathrm{prob}(D'(G_1(U_n)) = 1 \mid k = k_0)}_{= \cancel{\mathrm{prob}(D'(G_1(U_n)) = 1)}} \cdot \underbrace{\mathrm{prob}(k = k_0)}_{= \frac{1}{p(n)}}$$

independent!

$$= \sum_{k_0=0}^{p(n)-1} \mathrm{prob}(D(H_n^{k}) = 1 \mid k = k_0)$$

$$\mathrm{prob}(D'(U_{n+1}) = 1) = \frac{1}{p(n)} \sum_{k_0=0}^{p(n)-1} \mathrm{prob}(D(H_n^{k_0+1}) = 1 \cancel{\mid k=k_0})$$

$$\mathrm{prob}\big(D'(G_1(U_n)) = 1\big) - \mathrm{prob}\big(D'(U_{n+1}) = 1\big)$$

$$= \frac{1}{p(n)}\Big(\mathrm{prob}\big(D(\underset{\substack{\| \\ G(U_n)}}{H_n^0}) = 1\big) - \mathrm{prob}\big(D(\underset{\substack{\| \\ U_{p(n)}}}{H_n^{p(n)}}) = 1\big)\Big)$$

Thus: if D distinguishes $G(U_n)$ and $U_{p(n)}$

then D' distinguishes $G_1(U_n)$ and U_{n+1}. $\square$

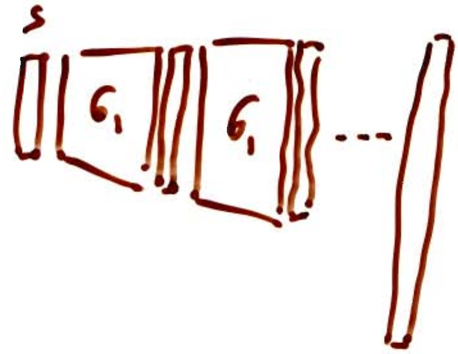
TiL & Matthias construction

G^{TM}

Input: $s \in \{0,1\}^n$

Output: $\sigma \in \{0,1\}^{p(n)}$.

1. ~~For~~ $s_n \leftarrow s$.
2. For $i = n+1 \dots p(n)$ do
3. $\quad s_i \leftarrow G_1(s_{i-1})$
4. Return $s_{p(n)}$.



$$\boxed{\begin{aligned} &G^{TM}(s) \\ &= G_1^{p(n)-n}(s) \end{aligned}}$$

Thm? G^{TM} is a prg. if G_1 is prg.

Ex Proof this! $\qquad H_n^k = G_1^{p(n)-n-k}(U_{n+k})$

But D'?... $\qquad \to H_n^0 = G^{TM}(U_n)$, $H_n^{p(n)-n} = U_{p(n)}$.

Proof (Yao, simple direction)

Assume that A is a prophet for the ensemble (X_n).

(Think of $X_n = G(U_n)$ for a generator G.)

Define a distinguisher D:

Input: α bit string

Output: 0 or 1.

1. If $A(\alpha, 1^n) = next_A(\alpha)$
   then return 1
   else return 0.

$$\text{prob}(D(X_n) = 1) = \text{prob}(A(X_n, 1^n) = next_A(X_n))$$

$$\text{prob}(D(U_{\ell(n)}) = 1) = \frac{1}{2}.$$

$$|\text{prob}(D(X_n)=1) - \text{prob}(D(U_{\ell(n)})=1)|$$

$$= |\underbrace{\text{prob}(A(X_n, 1^n) = next_A(X_n))}_{\geq \frac{1}{2} + \frac{1}{p(n)}} - \frac{1}{2}|$$

for infinitely many n and some polynomial p

$\geq \frac{1}{p(n)}$. But (X_n) and $(U_{\ell(n)})$ are assumed to be c.i.

↯ □

Thus c.i. from $(U_{\ell(n)})$ $\Rightarrow$ unpredictable.

Proof (Yao, unpredictable $\Rightarrow$ c.i. from $(U_{\ell(n)})$)

So assume to the contrary that D is a good distinguisher,

i.e. $$|\mathrm{prob}(D(X_n)=1) - \mathrm{prob}(D(U_{\ell(n)})=1)| > \frac{1}{p(n)}$$

for infinitely many n and some polynomial p.

Define the algorithm A:

Input: α bit string of length $\ell(n)$,
1^n.

Output: σ a guess for first unread bit of α.

1. Choose $k \in_R \{0, \dots, \ell(n)^{-1}\}$ uniformly random
2. Choose $\beta_{k+1}, \dots \beta_{\ell(n)} \in_R \{0,1\}$ uniformly random and independent.
3. If $D(\alpha_1 \dots \alpha_k \beta_{k+1} \dots \beta_{\ell(n)}) = 1$
   then Return β_{k+1}
   else Return $\overline{\beta_{k+1}}$

H_n^k = the first k bits of X_n, $U_{\ell(n)-k}$.

Observe: $H_n^0 = U_{\ell(n)}$, $H_n^{\ell(n)} = X_n$.

Heuristic: $H_n^{k+1} \sim H_n^k$ reflects 'prophets'.

Hybrids $= \ell(n)+1$ polynomial.

7.8.07
8

wlog.:

$$\text{prob}(D(X_n)=1) - \text{prob}(D(U_{\ell(n)})=1) > \frac{1}{p(n)}$$

for infinitely many n and some polynomial p.
(Otherwise flip the output of D.)

We want to show that

$$\text{succ}_A := \text{prob}(A(X_n, 1^n) = \text{next}_f(X_n))$$

$$> \frac{1}{2} + \frac{1}{\text{poly}(n)} \quad \boxed{= \frac{1}{2} + \frac{1}{\ell(n)}\frac{1}{p(n)}}$$

First split according to the chosen k. independent

$$\text{succ}_A = \underbrace{\frac{1}{\ell(n)}}_{= \text{prob}(k=k_0)} \sum_{k_0=0}^{\ell(n)-1} \text{prob}(\underbrace{A_{k_0}(X_n, 1^n)}) = X_n^{(k_0+1)} \cancel{| k=k_0})$$

if $D(\underbrace{X_n^{(1)} \ldots X_n^{(k)}} \underbrace{U_{\ell(n)}^{(k+1)} \ldots U_{\ell(n)}^{(\ell(n))}}) = 1$ H_n^k
answers $U_{\ell(n)}^{(k+1)}$
else it answers $1 - U_{\ell(n)}^{(k+1)}$

$$= \frac{1}{\ell(n)} \sum_{k_0} \Big(\text{prob}(D(H_n^{k_0}) = 1 \wedge (H_n^{k_0})^{(k_0+1)} = X_n^{k_0+1} \cancel{| k=k_0})$$

$$+ \text{prob}(D(H_n^{k_0}) = 0 \wedge (H_n^{k_0})^{(k_0+1)} \neq X_n^{k_0+1} \cancel{| k=k_0}) \Big)$$

$$H_n^{k_0}: \underbrace{X_n^{(1)} \ldots X_n^{(k_0)} \underbrace{U_{\ell(n)}^{(k_0+1)}}_{= X_n^{(k_0+1)}}}_{H_n^{k_0+1}} U_{\ell(n)}^{(k_0+2)} \ldots U_{\ell(n)}^{(\ell(n))}$$

7.8.07
3

The other would be:

$$\bar{H}_n^{k_0+1} := X_n^{(1)} \dots X_n^{(k_0)} \, \overline{X_n^{(k_0+1)}} \, U_{\ell(n)}^{(k_0+2)} \dots U_{\ell(n)}^{(\ell(n))} .$$

$$\text{prob}(D(H_n^{k_0}) = 1)$$

$$= \frac{1}{2} \left(\text{prob}(D(H_n^{k_0+1}) = 1) + \text{prob}(D(\bar{H}_n^{k_0+1}) = 1) \right)$$

Turning this upside down:

$$\text{prob}(D(\bar{H}_n^{k_0+1}) = 1) = 2\, \text{prob}(D(H_n^{k_0}) = 1) - \text{prob}(D(H_n^{k_0+1}) = 1)$$

Thus

$$\text{succ}_A = \frac{1}{2} \frac{1}{\ell(n)} \sum_{k_0} \Big(\text{prob}(D(H_n^{k_0+1}) = 1 \underbrace{\cancel{\wedge\, U = X_n^{k_0+1}}}_{\to \text{ a factor } 1/2}) + \text{prob}(D(\bar{H}_n^{k_0+1}) = 0 \cancel{\wedge\, U \neq X_n^{k_0+1}}) \Big)$$

$$= \frac{1}{2\ell(n)} \sum_{k_0} \Big(\text{prob}(D(H_n^{k_0+1}) = 1) + 1 - \text{prob}(D(\bar{H}_n^{k_0+1}) = 1) \Big)$$

$$= \frac{1}{2\ell(n)} \sum_{k_0} \Big(\text{prob}(D(H_n^{k_0+1}) = 1) + \underline{\underline{1}} - 2\, \text{prob}(D(H_n^{k_0}) = 1) + \text{prob}(D(H_n^{k_0+1}) = 1) \Big)$$

$$= \frac{1}{2} + \frac{1}{\ell(n)} \sum_{k_0} \Big(\text{prob}(D(H_n^{k_0+1}) = 1) - \text{prob}(D(H_n^{k_0}) = 1) \Big)$$

$$> \frac{1}{2} + \frac{1}{\ell(n)} \frac{1}{p(n)} \quad = \frac{1}{2} + \frac{1}{\ell(n)} \Big(\text{prob}(D(X_n) = 1) - \text{prob}(D(U_{\ell(n)}) = 1) \Big)$$

$\square$

Tuesday summary

8.8.07 (1)

Def pseudo-random $(X_n) \underset{\uparrow}{\sim} (U_{\ell(n)})$

pseudo-random generator c.i.

+ poly-time

+ $\ell(n) > n$.

Thm short becomes long [Hybrids]

Def unpredictable $A(X_n) = \mathrm{next}_A(X_n)$ difficult.

Thm unpredictable $\Leftrightarrow$ c.i. [Hybrids].

Outlook Wednesday (morning)

Connection to one-way functions

Examples

8.8.07
2

Proof $\exists$ prg $\Rightarrow$ $\exists$ one-way

$f: \{0,1\}^* \to \{0,1\}^*$
$(x,y) \mapsto G(x)$
$|x| = |y| \pm 1$

supposing that G is a prg with $\ell(n) = 2n$.

Clear: f is easy to compute.
So: assume f is not hard to invert.
Then there is an attacker (inverter) A with non-negligible success, i.e. there is a polynomial $p(n)$ and infinitely many n with

$$\text{prob}\big(A(f(U_n), \overset{1^n}{\overset{\shortparallel}{1^{|x|}}}) \in f^{-1} f(U_n) \big) > \frac{1}{p(n)}$$

Define Algorithm D
Input: $\alpha \in \{0,1\}^{2n}$.
Output: 0 or 1

1. Call the attacker to obtain a preimage: (x,y) with $\underset{\overset{\shortparallel}{f(x,y)}}{G(x)} = \alpha$.

If the attacker fails let $(x,y) \leftarrow 0^{2n}$.

2. If $G(x) = \alpha$ (attacker successful) then Return 1 else Return 0.

$$\operatorname{prob}\big(D(\underbrace{G(U_n)}_{\substack{\shortparallel \\ f(U_n,\, U_n')}}) = 1 \big)$$

$$= \operatorname{prob}\big(A(f(\underbrace{U_n, U_n'}_{U_{2n}})) \in f^{-1} f(U_n, U_n') \big)$$

$$> \frac{1}{p(n)} \text{ for infinitely many } n.$$

$$\operatorname{prob}\big(D(U_{2n}) = 1 \big) \le \frac{2^{n}}{2^{2n}} = \frac{1}{2^{n}}.$$

$|x|$

$|x|+|y|$ $\quad$ $|y|$.

$\# \{ G(x) \mid x \in \mathbb{B}^n \} \le 2^n,$

$\# \{ z \mid z \in \mathbb{B}^{2n} \} = 2^{2n}.$

$$\operatorname{prob}\big(D(G(U_n)) = 1 \big) - \operatorname{prob}\big(D(U_{2n}) = 1 \big)$$

$$> \frac{1}{p(n)} - \frac{1}{2^n} \underset{n \text{ large}}{>} \frac{1}{2p(n)}$$

↯ G is prg. $\square$

8.8.07
④

RSA

Choose p, q $n/2$-bit primes,
let $N = p \cdot q$,
$L = (p-1)(q-1)$,
$e < L$ coprime to L.

Then

$$f(x) = x^e \text{ in } \mathbb{Z}_N.$$

Consider $b(x)$ = least significant bit of x
$= \mathrm{bit}_0(x)$

Claim: If you have an algorithm to compute
$b(x)$ ~~the~~ given $f(x)$
then you can compute x!

Sketch: We have $y = f(x)$.
By assumption we can obtain $x_0 = b(x)$.

Case $x_0 = 0$

Then $x = 2x'$

$$\left.\begin{aligned} &\text{thus} \quad y = x^e = (2x')^e = 2^e x'^e \\ &\text{and so} \quad y' := x'^e = y/2^e. \end{aligned}\right\} \text{in } \mathbb{Z}_N.$$

Now ask $b(x')$.
But that is Bit 1 of x!

Case $x_0 = 1$ Then $-x = N - x$ is even... $\triangle$

Proof ($\exists$ one-way perm $\Rightarrow$ $\exists$ prg)

We have: f is one-way, $|f(x)| = |x|$,
f bijective.

b is a hard-core predicate for f.

Define $G(s) = \underbrace{f(s)}_{n \text{ bits}} \underbrace{b(s)}_{1 \text{ bit}}$.

Claim: G is a prg.

Clearly:
- $\ell(n) = n+1 > n$.
- G is poly-time.

Now suppose G is not a prg.

Then it is predictable,
ie. there exists a prophet A
such that

$$succ_A = prob(A(G(U_n)) = next_A(G(U_n)))$$

$$> \frac{1}{p(n)} + \frac{1}{2} \text{ for } \infty \text{ many } n$$
and some poly p.

Assume that A predicts not the last bit:

then $prob(A(\underbrace{f(U_n)}_{\doteq U_n'})) = next_A(\underbrace{f(U_n)}_{= U_n'}))$

$$= 1/2 . \not> \frac{1}{2} + \frac{1}{p(n)}$$

Thus A predicts the last bit:

$$\mathrm{succ}_A = \mathrm{prob}\left(A(f(U_n),\theta) = b(U_n) \right)$$

$$> \frac{1}{2} + \frac{1}{p(n)}$$ for ∞-many n and some p.

↯ b is hard-core for f.

$\square$

8.8.07

7

B1 Consider $g : \{0,1\}^3 \to \{0,1\}^6$

x	
000	001100
001	001110
010	010101
011	011011
100	101000
101	100101
110	110010
111	110011

(iii) Design a good prophet for g.

(iv) Transform it to a good distinguisher.

(v) Use the distinguisher $D(y) = \chi(\mathrm{wt}(y) \le 4)$ to derive a prophet as in the proof of YAO's theorem and compute its supposed and its actual advantage.

B2 (i) Repeat the proof that $\exists$ one-way permutations implies $\exists$ prg for one of the three examples.

(ii) ... for one-way functions that are (≤ 4)-to-1 and length-preserving.

B3 Complete the proof that the least significant bit of x is hard-core for $f(x) = x^e$ in $\mathbb{Z}_N$.

(i) assuming prob(...) $= 1$ for the algorithm outputting b.

(ii) assuming less...

B4 Make a mind map of the major topics of the course.

B1 8.8.07

(8)

(iii) $P_1(\alpha_0\alpha_1\alpha_2\alpha_3\alpha_4\alpha_5) = \alpha_0+\alpha_1+\alpha_2$

$\text{prob}(P_1(g(U_3)) = \text{next}_{P_1}(U_3)) = 1$

look at each of the 8 cases.

$P_2(\alpha) =$ Wait for a run of length 2, predict the opposite:

$j = \min\{i \mid \alpha_i = \alpha_{i+1}\}$

if $j<\infty$ then Return $\overline{\alpha_{j+1}}$

else Return a random bit.

$\text{prob}(P_2(g(U_3)) = \text{next}_{P_2}(g(U_3))) = \frac{1}{8}+\frac{1}{8}+\frac{1}{16}+\frac{1}{8}+0+\frac{1}{8}+\frac{1}{8}+\frac{1}{8}$

$= \frac{13}{16}.$

(iv) $D_1(\alpha) = \chi(P_1(\alpha) = \alpha_3)$

$D_2(\alpha) = \chi(P_2(\alpha) = \text{next}_{P_2}(\alpha))$

$\text{adv}_{D_1} = 1 - \frac{1}{2} = \frac{1}{2}$

$\text{adv}_{D_2} = \frac{13}{16} - \frac{32}{64} = \frac{52-32}{64} = \frac{20}{64}$

(v) Algorithm P_4

Input: $\alpha \in \{0,1\}^6$

Output: 0 or 1.

1. Choose $k \in_R \{0,\dots,5\}$.
2. Choose $\beta_{k+1},\dots,\beta_5 \in_R \{0,1\}$.
3. If $\text{wt}(\alpha_0\dots\alpha_k\beta_{k+1}\dots\beta_5) \le 4$

   then Return β_{k+1}

   else Return $\overline{\beta_{k+1}}$

8.8.07

9

$$\text{prob}(P_4(g(v_3)) = \text{next}_{P_4}(g(v_3)))$$

$$\geq \frac{1}{2} + \frac{1}{6} \cdot \frac{7}{64} = \frac{199}{384} \approx 0{,}518\ldots$$

$$\text{prob}(P_4(g(v_3)) = \text{next}_{P_4}(g(v_3)))$$

Wednesday summary

- one-way functions
- hard-core predicate / bit
- $\exists$ prg $\Rightarrow$ $\exists$ one-way fn.
- Example : RSA one-way? ~~An~~ permutation, with specific hard-core bit.
- Example : Blum, Blum, Shub (perm & hard-core)
- $\exists$ one-way permutation $\Rightarrow$ $\exists$ prg.
- OPEN QUESTION: $\exists$ one-way fn. $\Rightarrow$ $\exists$ one-way perm?
- DEEP THM: $\exists$ one-way $\Rightarrow$ $\exists$ prg.

- Protocol SUDOKU
  - $\rightarrow$ Round 1 : Provers and verifiers were honest
  - $\rightarrow$ Round 2 : Verifier honest, but provers arbitrary
  - $\rightarrow$ Round 3 : Provers honest, but verifier arbitrary
    $\rightarrow$ Verifier builds its simulator.

$BPP \subseteq IP$ Say M is a BPP-machine accepting $L \in BPP$.

Then

Paula	$\overset{x}{\longleftrightarrow}$	Victor
–		$M(x)$

$NP \subseteq IP$ Say $L \in NP$.

Then there exists a relation $R \in P$ consisting of pair (x, w) where $|w| \leq poly(|x|)$ and $x \in L \overset{\exists w: (x,w) \in R}{\Longleftrightarrow}$ and there is a det. poly-time machine M that accepts exactly elements of R.

Paula	$\overset{x}{\longleftrightarrow}$	Victor
Finds w such that $(x,w) \in R$	$\xrightarrow{w}$	Runs M to see whether $(x,w) \in R$ and accepts if so

Let $x \in L$:

$$prob(\langle P,V\rangle(x) = 1) = 1$$
$$prob(\langle P,V\rangle(x) \neq 1) = 0$$

Perfect completeness.

Let $x \notin L$:

$$prob(\langle B,V\rangle(x) \neq 0) = 0$$

Perfect soundness.

9.8.07
③

Proof (Protocol for GraphIsomorphism is perfect zero-knowledge)

Give P ~~(and any verifier~~ V^* / and / V.
What should the simulator M do?
Look at the protocol:

(G_0, G_1)

$\xrightarrow{\quad G' \quad}$

$\xleftarrow{\quad \sigma \quad}$

$\xrightarrow{\quad \psi \quad}$ verifier checks whether $\psi : G_\sigma \to G'$ is isomorphism

First idea for M:

select $\sigma = 0$, ψ permutation of V_G
create $G' = \psi\, G_\sigma$

Now $\text{prob}(M(x) = (G', \sigma, \psi))$

$$= \begin{cases} \text{prob}(M(x) = (G', 0, \psi)) & \text{if } \sigma = 0 \\ 0 & \end{cases}$$

better: $\text{prob}(M(x) = (G', 1, \psi)) = 0$

but $\text{prob}(\langle P, V\rangle(x) = (G', 1, \psi)) > 0$ if $G' = \psi G_1$.

Second trial:

<u>Algorithm M</u>

Input: G_0, G_1 two (isomorphic) graphs.
Output: view of a conversation.

1. Choose τ ~~σ~~ $\in_R \{0, 1\}$,
2. choose ψ $\in_R$: $V_{\sigma\tau} \to V_{\sigma\tau}$ randomly,
3. let $G' = \psi\, G_{\sigma\tau}$. Then $\psi : G_{\sigma\tau} \to G'$ is iso.
4. Output $(G', \sigma\tau, \psi)$.

~~Now: $\text{prob}(M(G_0, G_1) = (G', \sigma, \psi)) = \text{prob}(\langle P, V\rangle(G_0, G_1) =$~~

9.8.07

(4)

Now:

$m := \text{prob}(\, M(G_0, G_1) = (G', \sigma, \psi))$

$= \text{prob}(\, \langle P, V \rangle \, (G_0, G_1) = (G', \sigma, \psi)) =: pv.$

$m = \frac{1}{2} \, \frac{1}{(\# V_\sigma)!} \cdot 1$ say $V_0 = V_1 = \mathbb{N}_{<n}$.

$= \frac{1}{2n!}$ whenever (G', σ, ψ) is a legal protocol,

$pv = \frac{1}{n!} \cdot \frac{1}{2} \cdot 1 = \frac{1}{2n!}$ ie. $\psi : G_\sigma \to G'$ iso.

For short: our protocol is – what we call – perfect ~~zero~~ honest-verifier zero-knowledge.

General case? We have to deal with V^*.

Algorithm M^*

Input: G_0, G_1 two (isomorphic) graphs.

Output: a possible output of V^*.

1. Choose $\tau \in_R \{0, 1\}$ as a guess for V^*'s σ.
2. Choose $\psi : V_\tau \to V_\tau$ randomly,
3. let $G' = \psi \, G_\tau$ so that $\psi : G_\tau \to G'$ iso.
4. Ask $V^*(G') \to \sigma$.
5. If $\tau \neq \sigma$ Return $\perp$.
6. Return $V^*(G', \sigma, \psi)$.

Details: Ex?

$\triangle$

(C1) Prove Proposition 3.9. 19.8.07 (5)

As a starter amplify ~~that~~ interactive proof for GRAPH NONISOMORPHISM to soundness error

(i) $1/3$

(ii) $1/2^n$ where n = # vertices in the graphs G_0, G_1.

(iii) Go for the full proof of Proposition 3.9.

(C2) Find a (perfect) zero-knowledge proof for SQUARENESS, assuming that factoring is difficult.

$$L = \{ (N, z) \mid N = p \cdot q,\ p, q \equiv_4 3 \text{ prime}, \\ z = x^2 \text{ in } \mathbb{Z}_N \text{ for some } x \}$$

Hint: the prover first tells the verifier a square $b = a^2$.

(C3) Find a (perfect) zero-knowledge proof for PSEUDO-SQUARENESS assuming ...

$$L = \{ (N, z) \mid N = p \cdot q,\ p, q \equiv_4 3 \text{ prime}, \\ z \neq x^2 \text{ in } \mathbb{Z}_N \text{ for any } x \}$$

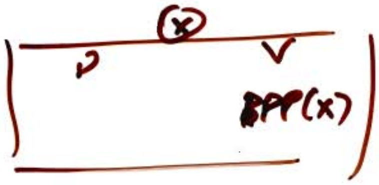
C2 partial answer (N, z)

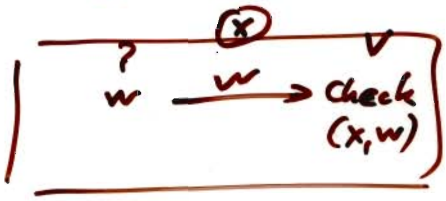
P $\boxed{z = x^2}$		V
Choose $a \in_R \mathbb{Z}_N$.		
Compute $b = a^2$.	$\xrightarrow{b}$	Choose $\sigma \in_R \{0,1\}$
	$\xleftarrow{\sigma}$	
Compute $w = x^{\sigma} a$		
	$\xrightarrow{w}$	Check $w^2 = z^{\sigma} \cdot b$ and accept iff true.

Thursday summary

10.8.07
①

- interactive TM & joint computation
- interactive proofs (generalized) IP
  - completeness error
  - soundness error
- BPP ⊂ IP & NP ⊂ IP





- GRAPH-NON-ISOMORPHISM
  interactive proof (ce = 0, se = ½)
- Rem: Public-coins vs. private coins (AM)
- perfect zero-knowledge
- statistical
- (computational)

} by <u>simulation</u>

- GRAPH-ISOMORPHISM
  zk proof (ce = 0, se = ½)

Quality of the simulator:

knowledge tightness: p_2.

$$\frac{t_{M^*}(x) - p_1(|x|)}{t_{V^*}(x)} < p_2(|x|)$$

for some polynomial p_1.

So clearly

$$P \subset BPP \subset PZK \subset SZK \subset ZK \subset IP \subset PSPACE$$

Believe: $\quad \neq ? \qquad\qquad =? \qquad \neq \qquad = \qquad =$

Amplify

→ Need auxiliary input to model that information is passed from one phase to the next phase during a multiple execution of a protocol.

Suppose we consider a Q-fold repetition (a polynomial) of a pair (P, V) and finally the new verifier will accept if in a α-fraction of the Q phases the original verifier accepted:

$$< P_Q , V_Q^{+\alpha} >$$

New completeness error?

$x \in L$: $\quad \text{prob}\left(<P_Q, V_Q^{+\alpha}>(x) \neq 1 \right)$

$$= \text{prob}\left(\dot{\exists} \underset{\#S > \alpha \cdot Q}{S \subset \{0,\dots,Q-1\}} : \begin{array}{l} \forall i \in S : <P^{(i)}, V^{(i)}>(x) = 1 \\ \forall i \notin S : <P^{(i)}, V^{(i)}>(x) \neq 1 \end{array} \right)$$

$$= \sum_S \prod_{i \in S} \text{prob}\left(<P,V>(x) = 1 \mid \dots \right) \prod_{i \notin S} \text{prob}\left(<P,V>(x) \neq 1 \right)$$

$$= \sum_{s=\lceil \alpha Q \rceil}^{Q} \binom{Q}{s} (1-ce)^{s} \, ce^{Q-s}$$

$$= \sum_{s=0}^{\lfloor (1-\alpha) Q \rfloor} \binom{Q}{s} (1-ce)^{Q-s} \, ce^{s}$$

$$\leq \exp\left(-2 \cdot Q \cdot (1-ce-\alpha)^2 \right)$$

↑ use Hoeffding inequality

New soundness error?

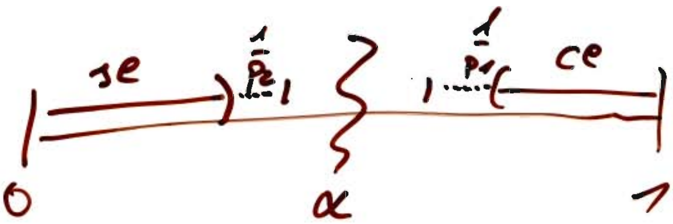
$x \notin L$: $\text{prob}\left(\langle P_Q, V_Q^{+\alpha} \rangle (x) \neq 0 \right)$

$$\leq \exp\left(-2Q \, (\alpha - se)^2 \right)$$

Must have for both to be small:

$$1 - ce - \alpha > \frac{1}{p_1(n)}$$

$$\alpha - se > \frac{1}{p_2(n)}$$



This leads to a proof of Prop. 3.9 that we can amplify interactive proofs.

But what about zero-knowledge?

Thm The protocol for G3C
is a (p.) zero-knowledge
interactive proof (if boxes exist).

Pf Completeness:
a 3-colorable graph
$$\mathrm{prob}(\langle P, V\rangle(x) \neq 1) = 0 !$$
So we even have perfect completeness.

Soundness:
Given any cheating prover B consider
the error
a not 3-colorable graph on n vertices
$$\mathrm{prob}(\langle B, V\rangle(x) \neq 0)$$
$$= 1 - \mathrm{prob}(\langle B, V\rangle(x) = 0)$$
$$\leq 1 - \frac{1}{\#E} \leq 1 - \frac{1}{n^2}$$

⌜ Repeating n^2 times brings it down to
$$\left(1 - \frac{1}{n^2}\right)^{n^2} \approx e^{-1} < \frac{1}{2}. ⌟$$

(Perfect) zero-knowledge:
Simulator M^*:
Guess an edge (u', w')
Choose keys and boxes $C_{u'}$ with $c_{u'}$
and $C_{w'}$ with $c_{w'}$
with $c_{u'} \neq c_{w'}$
and choose (keys and) boxes C_v with 0
for $v \in V \setminus \{u', w'\}$.
Ask: if $V^*((C_v)_{v \in V}) \neq (u', w')$ then retry (at most poly-times)
Return $V^*((C_v), (u', w'), (O_{u'}, c_{u'}, O_{w'}, c_{w'}))$

10.8.07
5

we would have to prove

$$\mathrm{prob}\,(\ M^*(\ \underset{\substack{\uparrow \\ \text{aux. input to } V^*}}{x, z}) = \alpha\ )$$

3-colorable graph

bitstring

$$= \mathrm{prob}\,(\ \langle P_{(y)}, V^*(z) \rangle\,(x) = \alpha\ ).$$

in case the boxes are perfect... Δ

Sketch ($\exists$ injective one-way $\Rightarrow$ $\underline{\exists \text{ bit commitment}}$)

more precise: the constructed protocol is a b.c..

① everything is randomized poly-time.

② perfectly binding: need only injectivity here!

Receiver sees $(f(s),\ b(s) \oplus v)$.

$f(s)$ determines s (though that cannot be found fast)

Thus $b(s)$ is determined

and so v is determined.

③ computationally hiding:

Distinguishing $(f(s), b(s))$

from $(f(s), b(s) \oplus 1)$

means ~~predicting~~ computing $b(s)$ from $f(s)$.

That cannot be done efficient

because b is hard-core. $\square$

10.8.07
6

Pf ($G3C \in \mathcal{ZK}$)

Perfect Completeness: c.e. = 0

Soundness: Suppose G is not 3-colorable.

$$\text{prob}(\langle B, V\rangle(G) = 1 (\neq 0)) \leq 1 - \frac{1}{\#E} < 1 - \frac{1}{n^2}.$$

Perfect binding!

Zero knowledge

Note that the simulator either
can find a 3-coloring
or the commitsments encode a non-3-coloring which in a actual transcript with the honest prover would never happen.

So assuming that G3C is difficult the simulator ~~not~~ always outputs wrong transcript.

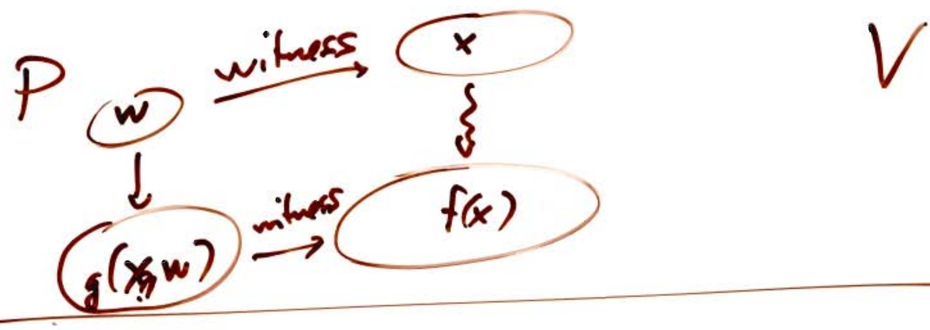
Yet, we only need that they are indistinguishable from real ones.

We will need Hybrids to break down distinguishing simulation from conversation to a bit commitment to 0 from a bit commitmentement to 1.

10.8.07
7

First: Assume $L \in NP$.
Then there exists a poly-time computable function f such that
$$f(x) \in G3C \iff x \in L.$$ KARP reduction

Let's try to set up a ZK proof for L.



$\boxed{\text{run a ZK proof for } f(x) \in G3C.}$

Claim: all standard transformations (say from L to SAT, to 3SAT, to G3C) allow the required poly-time witness transform g.

Completeness: even perfect completeness.

Soundness: Soundness error $\underset{(=)}{\leq}$ se(G3C) ✓

Zero knowledge: Simulator is just as before + first calculating $f(x)$ from x.
But a distinguisher would tell us only that $\langle P,V\rangle(\cancel{x}w)$ and $M(w)$ are different, yet we need that it sees a difference between '$g(x,w)$' < simulated conversa... Δ