Gems of TCS

Impagliazzo's Five Worlds

Sasha Golovnev

April 27, 2021

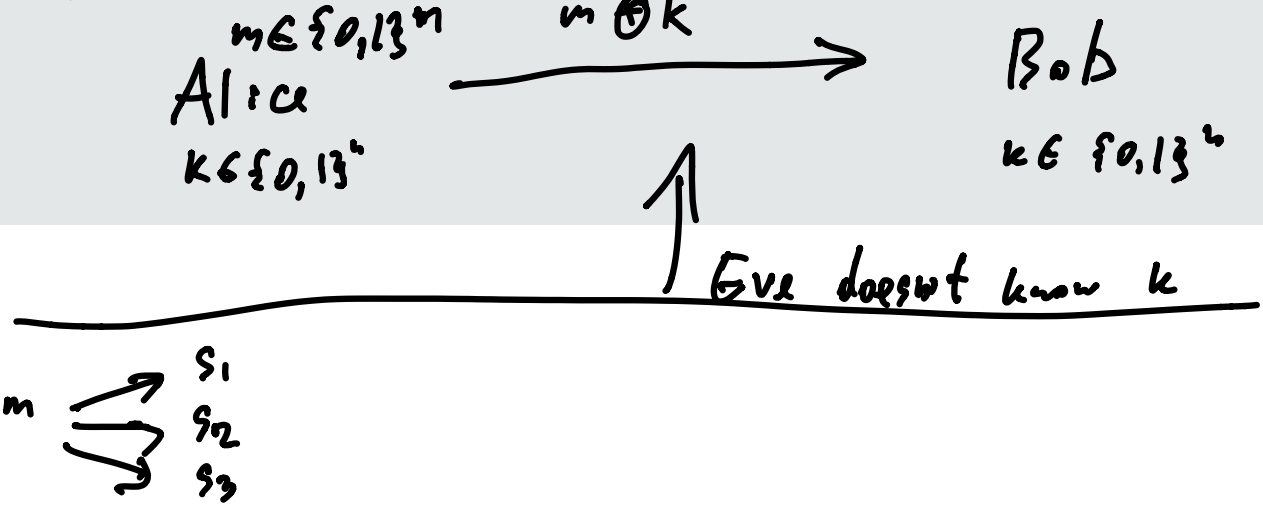
Cryptography

Three kinds of cryptography we've seen

Cryptography

Three kinds of cryptography we've seen

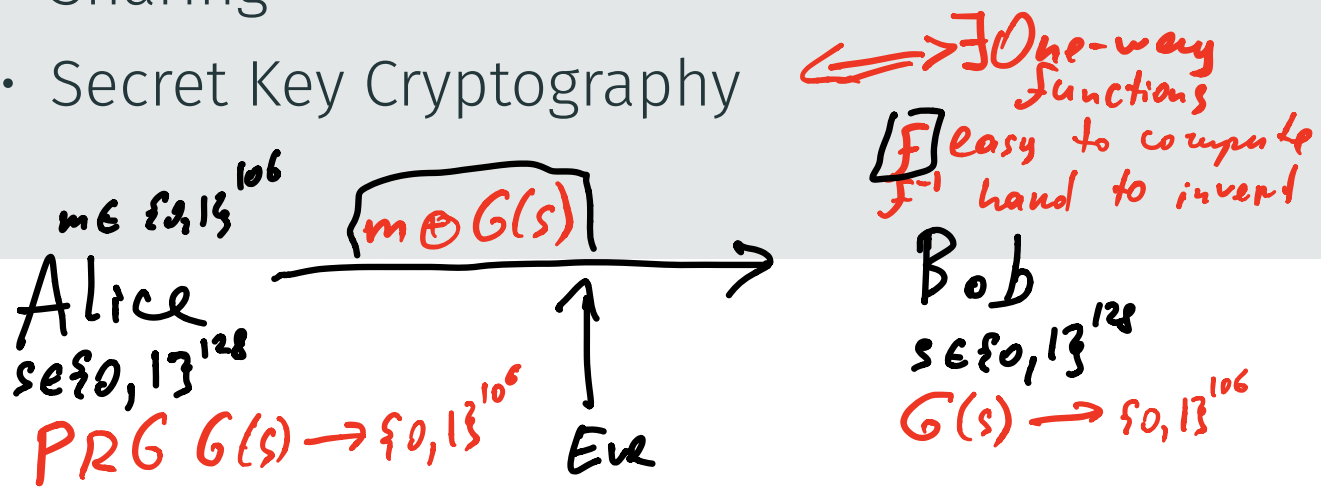
- Provably secure cryptography: OTP, Secret Sharing



Cryptography

Three kinds of cryptography we've seen

- Provably secure cryptography: OTP, Secret Sharing
- Secret Key Cryptography

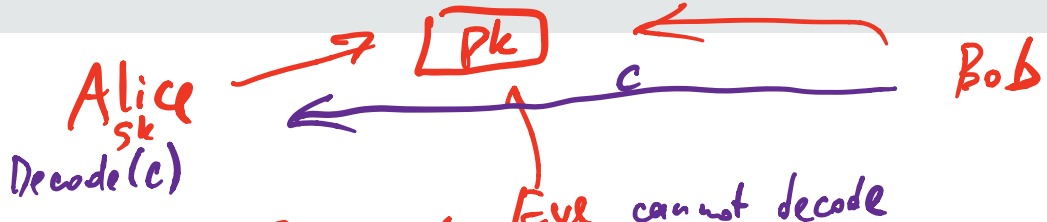


Cryptography

Three kinds of cryptography we've seen

1) Provably secure cryptography: OTP, Secret Sharing
2) Secret Key Cryptography
3) Public Key Cryptography

✓ OWF
✓ Factoring is hard
$\Leftrightarrow$ PKC assumptions



$\exists$ PKC $\Rightarrow$ $\exists$ OWF (SKC) $\Rightarrow$ P $\neq$ NP

Secret Key Cryptography

One-Way Function (OWF)

A function $f : \{0,1\}^* \to \{0,1\}^*$ is a one-way function if

Secret Key Cryptography

One-Way Function (OWF)

A function $f : \{0,1\}^* \to \{0,1\}^*$ is a one-way function if

- f is easy to compute: There exists a poly-time algorithm that computes f.

$f: \{0,1\}^n \to \{0,1\}^n$ bijection

Hard to invent : $x \in \{0,1\}^n$ $y = f(x)$, $\forall$ poly-time A

$Pr[A(y) \to x] < 1 - \frac{1}{n^2}$

$f: \{0,1\}^n \to \{0,1\}^*$ not nec. bijection

$F(00100) = 101$
$F(11001) = 101$
$F(01011) = 101$

A inverts if given $y = 101$ it finds any of the strings 00100, 11001, 01011

Secret Key Cryptography

∃ One-Way Function (OWF) ⇔ ∃ SKC

A function $f : \{0,1\}^* \to \{0,1\}$ is a one-way function if

- f is easy to compute: There exists a poly-time algorithm that computes f.
- f is hard to invert: For every poly-time algorithm $\mathcal{A}$ (and large enough n):

$$\Pr[x \in \{0,1\}^n : y = f(x), \mathcal{A}(y) = x', f(x') = y] \leq 1 - \frac{1}{n^2}.$$

Next: f_{Levin} is OWF iff ∃ OWF

Efficient OWF

Theorem

If there exists a OWF, then there exists a OWF computable in time $O(n^2)$.

Proof: $f : \{0,1\}^n \to \{0,1\}^n$ is OWF; f computed in time n^{100}

$g(x,y)$ s.t. $|y| = n$, $|x| = n^{100}$

$g(x,y) = (x, f(y))$

1. g is hard to invert. If one inverts g => one inverts f
2. g is easy to compute (n^2):

$n = n + n^{100}$

Run-time: $n^{100} + n^{100} = O(n) \leq n^2$

□

Levin's OWF

$M_1, M_2, M_3, \ldots\ldots$

$x \in \{0,1\}^n$

$$f_{Levin}(x) = M_1^{n^2}(x) \; M_2(x)^{n^2} \; M_3^{n^2}(x) \ldots\ldots \; M_n^{n^2}(x)$$

Theorem: $\exists$ OWF $\Rightarrow$ f_{Levin} is OWF

Proof: 1. Easy to compute: $n^2 \cdot n = O(n^3)$ – poly time

2. Hard to invert:

Assume $\exists$ OWF f comp in n^2; some M_K computes f.

$x \in \{0,1\}^n \quad n \geq K.$

$$f_{Levin}(x) = M_1^{n^2}(x) \; M_2^{n^2}(x) \ldots\ldots \; M_K^{n^2}(x) \ldots\ldots \; M_n^{n^2}(x)$$

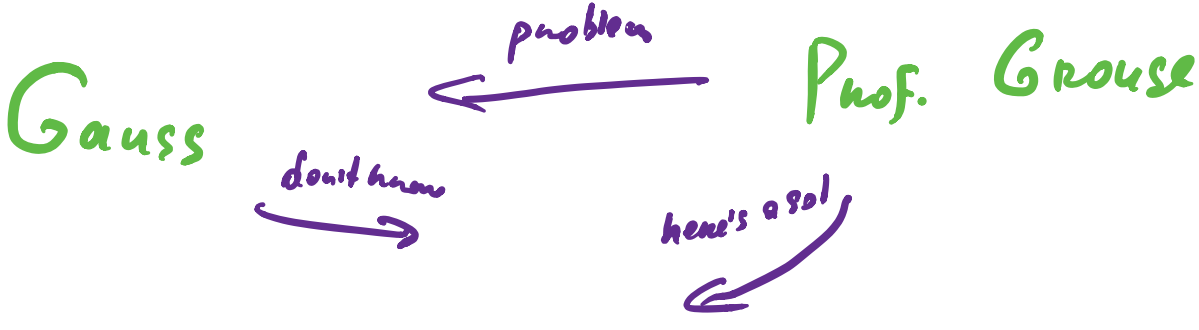
IF one invents f_{Levin} $\Rightarrow$ invent M_K $\Rightarrow$ invent f $\Rightarrow$ contradiction

□

Impagliazzo's Five Worlds

1995

Depending on which conjectures in complexity & cryptography are true/false, we live in one of 5 worlds



1. Algorithmics

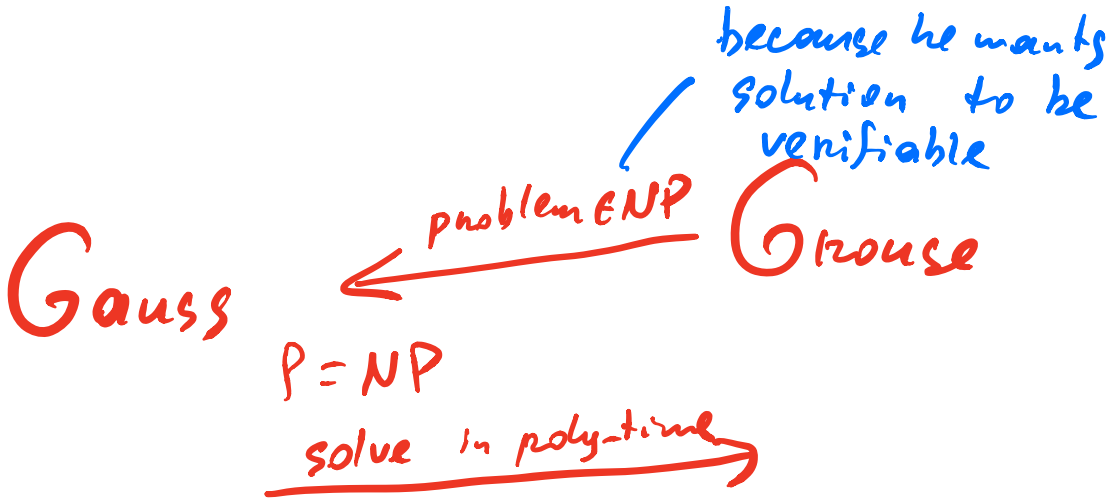
P = can be solved in poly-time

NP = can verify solution in poly time

Factoring

$P = NP$

- TSP can be solved in poly-time
- Most of optimization / economy / bioinformatics / can be solved in poly-time
- Programming languages: descriptive programming - what result to achieve
- AI/ML: train compute to do experts jobs
- No cryptography / no privacy


because he wants solution to be verifiable
problem ∈ NP
Grouse
Gauss
P = NP
solve in poly-time

2. Heuristics

Every NP-problem can be solved in poly-time for **most** inputs.

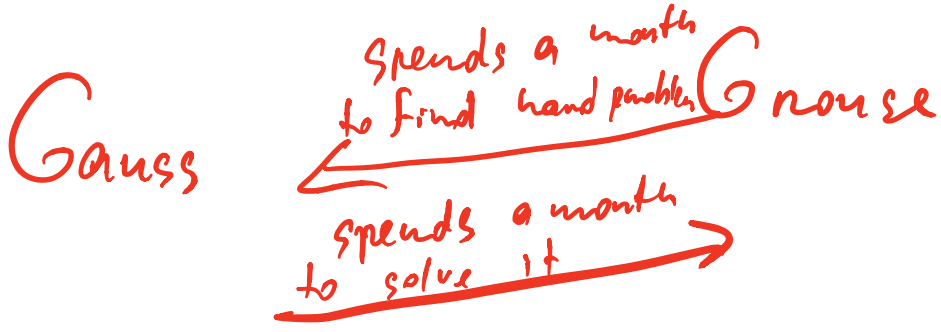
cannot solve it in poly-time on **all** inputs

$P \neq NP$ in theory

$P = NP$ in practice

SAT Crypto doesn't exist

Finding hard instances of NP-hard problems is itself a hard problem



3. Pessiland

$P \neq NP$

NP problems cannot be solved for most inputs

Still Cryptography still doesn't exist

- even in practice we cannot solve problems economy / scheduling / ML...
- don't have security

Gauss — hard problem → Gnome

Gauss — don't know

Gnome — me neither

4. Minicrypt

$P \neq NP$

$\Uparrow$

$\exists OWF$ ($\exists$ SKC)

- we can't solve NP-problems even for most inputs
- we have problems that are so hard that

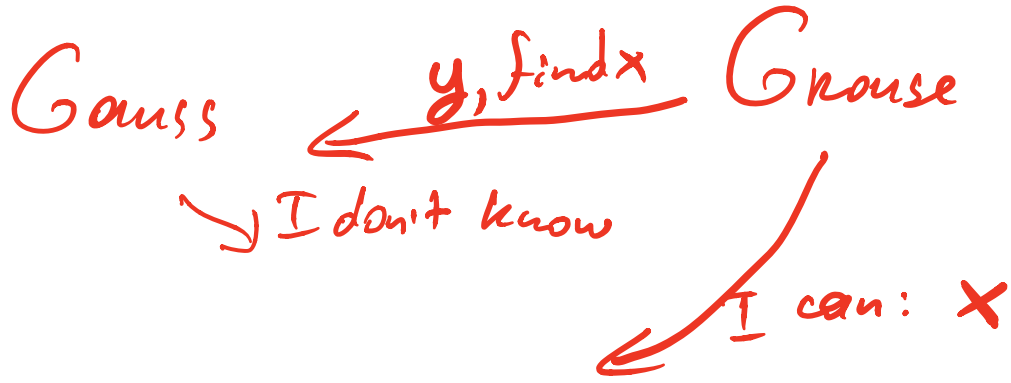
  f: compute efficiently

  f^{-1}: cannot invert efficiently

- SKC cryptography

- No PKC, no anonymous digital money

OWF f: $x \Rightarrow y = f(x)$



5. Cryptomania

$\exists$ PKC $\Rightarrow$ $\exists$ OWF ($\exists$ SKC)

$\Rightarrow$ P $\neq$ NP

- we cannot solve NP-problems efficiently even on most inputs

- Alice — PK — Bob — c — Alice — SK

— Anonymous digital money.....