Nov10

Monday, November 9, 2020 9:45 PM

Idea of collision algorithm

If I make two lists in a finite set S

$$\left.\begin{array}{l} x_1, x_2, \ldots, x_n \\ y_1, y_2, \ldots, y_n \end{array}\right\} \text{likely } x_i = y_j \text{ when } n = O(\sqrt{\#S})$$

Used this

$$\begin{array}{l} x_i = g^{\alpha_i} \\ y_j = h g^{\beta_j} \end{array} \Longrightarrow \begin{array}{l} x_i = y_j \text{ then} \\ \log_g h = \alpha_i - \beta_j \end{array} \quad \checkmark$$

Remarks

① Worse than Baby steps giant steps alg from weeks ago.

- * Longer lists ← $O(\sqrt{p-1})$
- * Fast powering
- * Only probably right.

⇝ interesting that BSGS not that much better than random lists.

* ② Both have HUGE memory constraints.

Today: Describe a collision algorithm that doesn't need save lists.

Idea

Compare x_i, y_i & continue w/o saving them.

For this to work we have to be clever about choices of x_i, y_i (relative to eachother).

Pollard's ρ method

Basic idea:

S = finite set

$f: S \longrightarrow S$ f does a good job of "randomly mixing elts of S"

Fix some $x \in S$ Consider

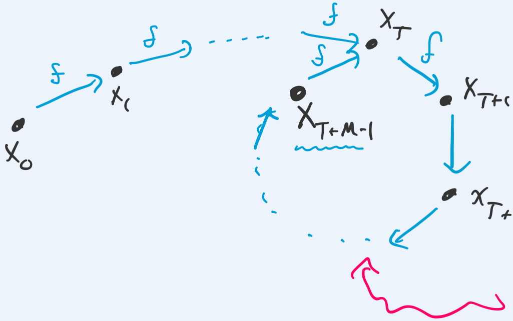
$x_0 = x$. $x_1 = f(x)$, $x_2 = f\circ f(x)$, . . .

$$x_i = f(x_{i-1}) = \underbrace{f\circ f\circ \cdots \circ f}_{i\text{-times.}}(x)$$

Defn Such a function/sequence is called a discrete dynamical system.

$x_0, x_1, x_2, \ldots$ is called the orbit of x $\mathcal{O}_f^+(x)$

Picture



S is finite

$\Downarrow$

eventually overlap

$\Downarrow$

get stuck in this loop

Rmk looks like a ρ.

Eventually get

$$\left.\begin{aligned} x_T &= x_{T+M} \\ \text{so } x_{T+1} &= x_{T+M+1} \\ &\vdots \end{aligned}\right\} \text{repeating}$$

Def^n

$x_0, x_1, \dots, x_{T-1}$ tail of $O_f^+(x)$

$x_T, x_{T+1}, \dots, x_{T+M-1}$ loop of $O_f^+(x)$

$\rightsquigarrow$ T = # elts in loop

M = # elts in tail.

$T = \left(\begin{array}{l}\text{largest int s.t.} \\ x_{T-1} \text{ appears once } O_f^+(x)\end{array}\right)$

$M = \left(\begin{array}{c}\text{smallest int s.t.} \\ x_{T+M} = x_T\end{array}\right)$

Idea

- * $T+M$ is usually small $(O(\sqrt{\#S}))$
- * $x_{T+M} = x_T$ $\rightsquigarrow$ Found a collision.

Problem

* Still seem to need a list $x_0, x_1, \ldots, x_{T+M}$ to notice $x_{T+M} = x_T$.

(can't know T & M yet).

Trying to avoid making big lists........

Let's get around this

* Algorithm to detect a collision w/o storing all values

As we compute

① * $x_0, x_1, x_2, \ldots$.

Also compute

② * $\underbrace{y_0, y_1, y_2, \ldots}$

$\hookrightarrow$ $y_0 = x$. $y_1 = f \circ f(x)$. $y_2 = f \circ f \circ f \circ f(x)$

$\ldots$ $y_i = f \circ f(y_{i-1})$.

Notice $y_i = x_{2i}$ pf/ Automatic.

③ At each step check if $x_i = y_i$. If yes, done!

Upshot

Only need to remember x_i, y_i @ any step. No LISTS!

Question:

Does this work? yes How fast? T+M steps. }

Lemma $j > i$

$$x_i = x_j \iff \underline{i \ge T} \ \& \ \underline{j \equiv i \bmod M}.$$

pf/ If $j = i + kM$ & $i \ge T \leftarrow x_i$ in loop

"⇐" ⇒ gone around loop k times ✓

"⇒" Need x_i in loop. Go around integer # times

In action When does $\underline{x_i = y_i = x_{2i}}$ <T+M steps.

By lemma ⟺ $i \ge T$ & $\underbrace{2i \equiv i \bmod M}$

$$\hookrightarrow m \mid i(2-1) \Rightarrow m \mid i$$

So when i is first multiple of M in

$$T, T+1, T+2, \ldots T+M-1 \}$$

have $x_i = x_{2i}$

This always happens b/c w M consecutive #s one of them must be 0 mod M.

Say $x_i = x_{2i}$ for some $i < T+M$

Found a collision before finishing loop!!

Rmk useless if $T+M$ is HUGE.

In fact, ... $T+M \approx \sqrt{N}$

↓

Theorem (Pollard's ρ method)

Let S a finite set $\# S = N$.

Let $f : S \longrightarrow S$ be a function.

$x \in S$ an initial point w/ orbit $O_f^+(x) = x_1, x_2, \dots$

and tail size T & loop size M.

① $x_{2i} = x_i$ some $i < T+M$.

② If f is sufficiently random then $(N \gg 0)$.

$$E(T+M) \approx 1.2533\sqrt{N}$$

So we will find a collision in $O(\sqrt{N})$.

Pf/ ① Discussion above.

② (sketch)

First k values of $O_f^+(x)$, $x_0, x_1, \dots, x_{k-1}$

Prob that are all different. (i.e. $k < T+M$).

$$\Pr\left(\begin{matrix} x_0, \dots, x_{k-1} \\ \text{are different} \end{matrix}\right) = \prod_{i=1}^{k-1} \Pr\left(\begin{matrix} x_i \neq x_j \\ j=0, \dots, i-1 \end{matrix} \middle| \begin{matrix} x_0, \dots, x_{i-1} \\ \text{are different} \end{matrix}\right)$$

$$= \prod_{i=1}^{k-1} \left(\frac{N-i}{N} \right) = * \qquad 1-\frac{i}{N} \approx e^{i/N}$$

HW 9

t small, $1-t \approx e^{-t}$. In practice $k \approx \sqrt{N}$, $N >> 0$

so i/N small

$$* \approx \prod_{i=1}^{k-1} e^{-i/N} = e^{-(1+2+\dots+k-1)/N} \approx *$$

$$1+2+\dots+k-1 = \frac{k(k-1)}{2} = \frac{k^2-k}{2} \approx \frac{k^2}{2}$$

$$\Pr(x_0, \dots, x_{k-1} \text{ dist}) \approx \underline{e^{-k^2/2N}}.$$

First match is $T+M$.

Want $\Pr(x_k$ is first match$)$. $* =$

① $\Pr(x_k$ is a match $\mid \underline{x_0, \dots, x_{k-1}}$ distinct$) = k/N$

$*$ $\Pr(x_k$ is a match & $x_0, \dots, x_{k-1}$ are distinct$)$

$$= \Pr(x_k \text{ match} \mid x_0, \dots x_{k-1} \text{ dist}) \cdot \underline{\Pr(x_0, \dots, x_{k-1}} \text{ dist})$$

$$\approx \frac{k}{N} \cdot e^{-k^2/2N}$$

Showed $\boxed{\Pr(T+M=k) = \frac{k}{N}\cdot e^{-k^2/2N}}$

Rmk Crucial that f is random b/c we are treating the x_i's as random wrt eachother.

$$E(T+M) = E(\text{first match})$$

$$= \sum_{k=1}^{\infty} k\cdot \Pr(T+M=k)$$

$$\approx \sum_{k=1}^{\infty} \frac{k^2}{N}\cdot e^{-k^2/N} \quad \longleftarrow$$

what is this as a fn of N?

Lemma

Let $F(t)$ be a functions w/ a continuous derivative. from $\mathbb{R} \longrightarrow \mathbb{R}$.

Suppose $\int_0^\infty F(t)\,dt$ converges.

$$\Longrightarrow \sum_{k=1}^{\infty} F\left(\frac{k}{n}\right) \approx n\int_0^\infty F(t)\,dt. \Bigg\}$$

Lemma $\Longrightarrow$ Theorem

$F(t) = t^2\cdot e^{t^2/2}$.

$F(k/\sqrt{N}) = \frac{k^2}{n} e^{k^2/2N}$

$E(T+M) = \sum \frac{k^2}{N} e^{k^2/2N}$

$$= \sum F\left(k/\sqrt{N}\right) \quad \sqrt{\pi/2}$$

Lemma $$= \sqrt{N}\int_0^\infty t^2 e^{t^2/2}\,dt$$

$$= 1.2533\sqrt{N}$$

Proof thm

Proof of Lemma

$$\int_0^A F(t)\,dt = \lim_{n\to\infty} \sum_{k=1}^{An} F(k/n)\cdot\frac{1}{n}$$

so $n >> 0$

$$\longrightarrow n\int_0^A F(t)\,dt \approx \sum_{k=1}^{An} F(k/n) \longleftarrow$$

Let $A \longrightarrow \infty$

do bookkeeping

Using this to attack DLP

Solve $g^x = h$ in $\mathbb{F}_p^*$

Idea $f : \mathbb{F}_p^* \longrightarrow \mathbb{F}_p^* \leftarrow$ which is best

s.t. $x_i = g^{\alpha_i} h^{\beta_i}$

Then $x_i = x_j \rightsquigarrow g^{\alpha_i} h^{\beta_i} = g^{\alpha_j} h^{\beta_j}$

$$\longrightarrow g^{\alpha_i - \alpha_j} = h^{\beta_j - \beta_i}$$

so $\alpha_i - \alpha_j \equiv (\beta_j - \beta_i) \log_g h.$ mod $p-1$

Almost solve DLP

If $\underset{\substack{\shortparallel \\ g}}{\gcd}(\beta_j - \beta_i, p-1) = 1$ done.

~~E~~ so g solutions to

$$(\beta_j - \beta_i)\underset{\substack{\uparrow \\ \text{one of these is } \log_g h.}}{x} \equiv \alpha_i - \alpha_j \quad \text{mod } p-1$$

As in HW7 #9 enumerate them & check which one is right.

Pollard's idea

* simple enough to compute.
* Random enough to work.

$$\nearrow f(x) = \begin{cases} g \cdot x & 0 \leq x < p/3 \\ x^2 & p/3 \leq x < 2p/3 \\ x \cdot h & 2p/3 \leq x < p. \end{cases} \quad \text{mod } p$$

Certainly $x_i = g^{\alpha_i} h^{\beta_i}$ $\forall$ i

Also need to remember α_i, β_i at any given step.
But they can be computed in terms of α_{i-1}

& β_{i-1} as you compute x_i.

$$x_{i-1} = g^{\alpha_{i-1}} h^{\beta_{i-1}}$$

$$x_i = \begin{cases} g^{\alpha_{i-1}+1} h^{\beta_{i-1}} & x < p/3 \\ g^{2\alpha_{i-1}} h^{2\beta_{i-}} & p/3 \le x < 2p/3 \\ g^{\alpha_{i-1}} h^{\beta_{i-1}+1} & 2p/3 \le x < p. \end{cases}$$

So
$$\alpha_i = \left.\begin{cases} \alpha_{i-1}+1 & x < p/3 \\ 2\alpha_{i-1} & p/3 \le x < 2p/3 \\ \alpha_{i-1} & 2p/3 \le x \end{cases}\right\} \underset{\underset{\text{Fermat.}}{\uparrow}}{\bmod p-1}$$

β_i similar.

Similar way we can compute y_i

$$y_0 = 1 \qquad y_{i+1} = f(f(y_i))$$

$$y_i = g^{\gamma_i} \cdot h^{\delta_i}$$

At each step

$$\Big((\underset{\substack{\uparrow \\ \text{in } \mathbb{F}_p}}{x_i}, \underset{\substack{\uparrow\uparrow \\ \bmod p-1}}{\alpha_i, \beta_i}), \ (\underset{\substack{\uparrow \\ \text{in } \mathbb{F}_p}}{y_i}, \underset{\substack{\uparrow\uparrow \\ \bmod p-1}}{\gamma_i, \delta_i}) \Big) f \circ f$$

if

$(x_{i+1}, \alpha_{i+1}, \beta_{i+1})$ $(y_{i+1}, \gamma_{i+1}, \delta_{i+1})$

If I get a match

$$g^{\alpha_i} h^{\beta_i} = g^{\gamma_i} h^{\delta_i} \mod p$$

$\rightarrow u = \alpha_i - \gamma_i \quad v = \delta_i - \beta_i$

$\log_g h$ solves $\underbrace{vx = u \mod p-1}$

→ compute all solns & see which one works!

Idea: Find / use Extended Euclidean alg. Rest are $\frac{p-1}{g}$ multiples away } HW #7

Perfect Secrecy

Idea: ciphertext gives no information about plaintext. Formulate this probabilistically.

$$\Omega = \mathcal{M} \times \mathcal{C} \times \mathcal{K}$$

3 random variables $M(m, c, k) = m$

M = returns the message.

C = returns the cipher

K = return the key.

Get density functions f_M, f_C, f_K.

$$f_M(m) = \Pr(M=m)$$

Get joint density functions

$$f_{M,C}(m,c) = \Pr\left(\text{message is } m \text{ \& cipher is } c\right)$$

$$\longrightarrow f_{M|C}(m|c) = \Pr\left(\begin{matrix}\text{message is } m \\ \text{if cipher is } c\end{matrix}\right)$$

Def$^{\underline{n}}$ A cryptosystem has perfect secrecy if

$$f(m|c) = f(m) \quad \forall \begin{matrix} m \in \mathcal{M} \\ c \in \mathcal{C} \end{matrix}$$

the probability that the message is m is indep of ciphertext.

Bayes' $$f(m|c) \cdot f(c) = f(c|m) \cdot f(m)$$

Perfect Secrecy $$\Longleftrightarrow f(c) = f(c|m) \quad \forall \begin{matrix} m \in \mathcal{M} \\ c \in \mathcal{C} \end{matrix}$$

Any cipher is equally likely indep of plaintext.

Observation K, M indep. But K, M, C not.

$$\boxed{f_K, f_M \text{ determines } f_C.}$$

why?

Fix $k \in \mathcal{K}$.

$e_k : \mathcal{M} \longrightarrow \mathcal{C}$

$d_k : \mathcal{C} \longrightarrow \mathcal{M}$

$$f(c|k) = \Pr(C=c \mid K=k) = \Pr(M = \underline{d_k(c)}) = f_M(d_k(c))$$

So $f_C(c) = \Pr(C=c) \underset{\text{Avg}}{=} \sum f_{C|K}(c|k) f_K(k)$

$$= \sum f_M(d_k(c)) \cdot f_K(k)$$

depends on f_M & f_K.