Birthday paradox and birthday attacks

Example 211. (birthday paradox) Among $n=35$ people (a typical class size; no leaplings), how likely is it that two have the same birthday?

Solution.

$$1-\left(1-\frac{1}{365}\right)\left(1-\frac{2}{365}\right)\left(1-\frac{3}{365}\right)\cdots\left(1-\frac{34}{365}\right)\approx 0.814$$

If the formula doesn't speak to you, see Section 8.4 in our book for more details or checkout:

https://en.wikipedia.org/wiki/Birthday_problem

Comment. For $n=50$, we get a 97.0% chance. For $n=70$, it is 99.9%.

Comment. In reality, birthdays are not distributed quite uniformly, which further increases these probabilities. Also note that, for simplicity, we excluded "leaplings", people born on February 29.

How is this relevant to crypto, and hashes in particular?

Think about people as messages and birthdays as the hash of a person. The birthday paradox is saying that "collisions" occur more frequently then one might expect.

Example 212. Suppose M is large ($M=365$ in the birthday problem) compared to n. The probability that, among n selections from M choices, there is no collision is

$$\left(1-\frac{1}{M}\right)\left(1-\frac{2}{M}\right)\cdots\left(1-\frac{n-1}{M}\right)\approx e^{-\frac{n^2}{2M}}.$$

Why? For small x, we have $e^x\approx 1+x$ (the tangent line of e^x at $x=0$). Hence, $\left(1-\frac{k}{M}\right)\approx e^{-k/M}$ and

$$\left(1-\frac{1}{M}\right)\left(1-\frac{2}{M}\right)\cdots\left(1-\frac{n-1}{M}\right)\approx e^{-1/M}e^{-2/M}\cdots e^{-(n-1)/M}=e^{-(1+2+\ldots+(n-1))/M}=e^{-\frac{n(n-1)}{2M}}$$

In the last step, we used that $1+2+\ldots+(n-1)=\frac{n(n-1)}{2}$. Finally, $e^{-\frac{n(n-1)}{2M}}\approx e^{-\frac{n^2}{2M}}$.

Decent approximation? For instance, for $M=365$ and $n=35$, we get 0.813 for the chance of a collision (instead of the true 0.814).

Important observation. If $n\approx\sqrt{M}$, then the probability of no collision is about $e^{-1/2}\approx 0.607$. In other words, a collision is quite likely!

In the context of hash functions, this means the following: if the output size is b bits, then there are $M=2^b$ many possible hashes. If we make a list of $\sqrt{M}=2^{b/2}$ many hashes, we are likely to observe a collision.

For collision-resistance, the output size of a hash function needs to have twice the number of bits that would be necessary to prevent a brute-force attack.

Practical relevance. This is very important for every application of hashes which relies on collision-resistance, such as digital signatures.

For instance, think about Eve trying to trick Alice into signing a fraudulent contract m. Instead of m, she prepares a different contract m' that Alice would be happy to sign. Eve now creates many slight variations of m and m' (for instance, by varying irrelevant things like commas or spaces) with the hope of finding $\tilde{m}$ and $\tilde{m}'$ such that $H(\tilde{m})=H(\tilde{m}')$. (Why?!!)

Armin Straub
straub@southalabama.edu

Example 213. (chip based credit cards) Modern chip based credit cards use digitial signatures to authenticate a payment.

How? The card carries a public key, which is signed by the bank, so that a merchant can verify the public key. The card then signs a challenge from the merchant for authentication. The private key used for that is not even known to the bank.

Note that all of this can be done offline, without needing to contact the bank during the transaction.

https://en.wikipedia.org/wiki/EMV

There's an interesting and curious story made possible by the fact that, around 2000, banks in France used 320 bit RSA (chosen in the 80s and then not fixed despite expert advice) for signing the card's public key:

https://en.wikipedia.org/wiki/Serge_Humpich

Comment. For contrast, the magnetic stripe just contains the card information such as card number.

Comment. This also leads to interesting questions like: can we embed a private key in a chip (or code) in such a way that an adversary, with full access to the circuit (or code), still cannot extract the key?

https://en.wikipedia.org/wiki/Obfuscation#White_box_cryptography

A digital signature is like a hash, which can only be created by a single entity (using a private key) but which can be verified by anyone (using a public key).

As one might expect, a symmetric version of this idea is also common:

Example 214. (MAC) A **message authentication code**, also known as a **keyed hash**, uses a private key k to compute a hash for a message.

Like a hash, a MAC provides integrity. Further, like a digital signature, it provides authenticity because only parties knowing the private key are able to compute the correct hash.

Comment. On the other hand, a MAC does not offer non-repudiation because several parties know the private key (whether non-repudiation is desirable or undesirable depends on the application). Hence, it cannot be proven to a third party who among those computed the MAC (and, in any case, such a discussion would make it necessary to reveal the private key, which is usually unacceptable).

From hash to MAC. If you have a cryptographic hash function H, you can simply produce a MAC $M_k(x)$ (usually referred to as a HMAC) as follows:

$$M_k(x) = H(k, x)$$

This seems to work fine for instance for SHA-3. On the other hand, this does not appear quite as secure for certain other common hash functions. Instead, it is common to use $M_k(x) = H(k, H(k, x))$. For more details, see:

https://en.wikipedia.org/wiki/Hash-based_message_authentication_code

From ciphers to MAC. Similarly, we can also use ciphers to create a MAC. See, for instance:

https://en.wikipedia.org/wiki/CBC-MAC

Elliptic curve cryptography

The idea of Diffie–Hellman (used, for instance, in DH key exchange, ElGamal or DSA) can be carried over to algebraic structures different from multiplication modulo p.

Recall that the key idea is, starting from individual secrets x, y, to share g^x, g^y modulo p in order to arrive at the joint secret $g^{xy} \pmod p$. That's using multiplication modulo p.

One important example of other such algebraic structures, for which the analog of the discrete logarithm problem is believed to be difficult, are elliptic curves.

https://en.wikipedia.org/wiki/Elliptic_curve_cryptography

Comment. The main reason (apart from, say, diversification) is that this leads to a significant saving in key size and speed. Whereas, in practice, about 2048bit primes are needed for Diffie–Hellman, comparable security using elliptic curves is believed to only require about 256bits.

For a beautiful introduction by Dan Boneh, check out the presentation:

https://www.youtube.com/watch?v=4M8_Oo7lpiA

Points on elliptic curves

An **elliptic curve** is a (nice) cubic curve that can (typically) be written in the form

$$y^2 = x^3 + ax + b.$$

A point (x, y) is on the elliptic curve if it satisfies this equation. Each elliptic curve also contains the special point O ("the point at ∞"). [O will act as the neutral element when "adding points"]

Advanced comment. Sometimes it is useful (or necessary) to consider elliptic curves defined by more general cubic equations such as $y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$ (however, in most cases, a linear change of variables can transform this equation into the simpler form $y^2 = x^3 + ax + b$ mentioned above).

Example 215. Determine some points (x, y) on the elliptic curve E, described by

$$y^2 = x^3 - x + 9.$$

Solution. We can try some small values for x (say, $x = 0$, $x = 1$, $x = 2$, ...) and see what y needs to be in order to get a point on the elliptic curve. For instance, for $x = 1$, we get $x^3 - x + 9 = 9$ which implies that $(1, \pm 3)$ are points on the elliptic curve.

Doing so, we find the integral points $(0, \pm 3), (\pm 1, \pm 3)$.

On the other hand, for $x = 2$, we get $x^3 - x + 9 = 15$ which implies that $(2, \pm\sqrt{15})$ are points on the elliptic curve. However, for cryptographic purposes, we are usually not interested in such irrational points.

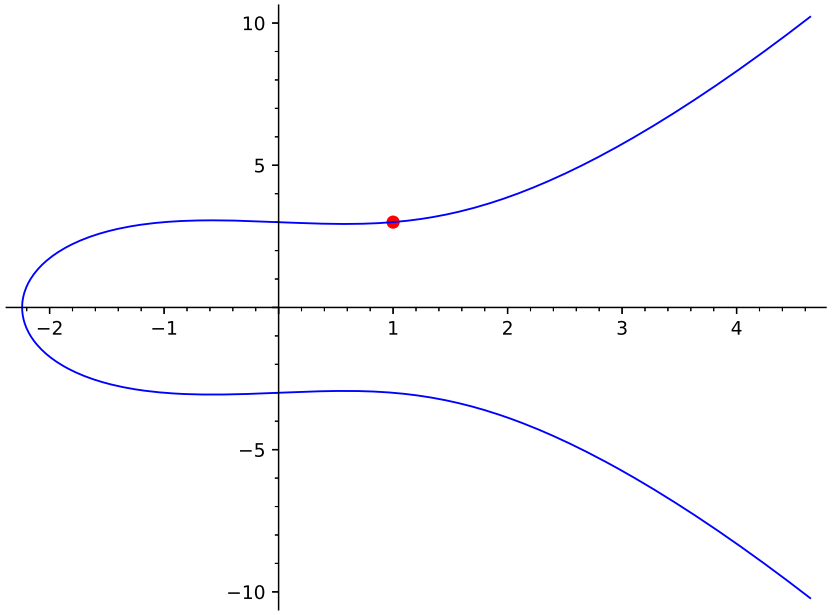
Much less obvious rational points include $(35, 207)$ or $\left(\frac{1}{36}, \frac{647}{216}\right)$ (see Example 217).

Comment. In general, it is a very difficult problem to determine all rational points on an elliptic curve, and lots of challenges remain open in that arena.

Example 216. Plot the elliptic curve E, described by $y^2 = x^3 - x + 9$ and mark the point $(1,3)$.

Solution. We let Sage do the work for us:

```
>>> E = EllipticCurve([-1,9])
>>> E.plot() + E(1,3).plot(pointsize=50, rgbcolor=(1,0,0))
```



Adding points on elliptic curves

Note. Simply adding the coordinates of two points P and Q on an elliptic curve will (almost always) not result in a third point on the elliptic curve. However, we will define a more fancy "addition" of points, which we will denote $P \boxplus Q$, such that the $P \boxplus Q$ is on the elliptic curve as well.

Given a point $P = (x, y)$ on E, we define $-P = (x, -y)$ which is another point on E.

Let us introduce an operation $\boxplus$ in the following geometric fashion: given two points P, Q, the line through these two points intersects the curve in a third point R.

We then define $P \boxplus Q = -R$.

We remark that $P \boxplus (-P)$ is the point O "at ∞". That's the neutral (zero) element for $\boxplus$.

How does one define $P \boxplus P$? (Tangent line!)

Remarkably, the "addition" $P \boxplus Q$ is associative. (This is not obvious from the definition.)

Using $\boxplus$, we can construct new points: for instance, $(0,3) \boxplus (1,-3) = (35, 207)$ as we will verify in the next example.

Easier to verify (but not producing anything new) is $(0,3) \boxplus (1,3) = (-1,-3)$.

Example 217. Consider again the elliptic curve E, described by $y^2 = x^3 - x + 9$.

(a) Determine $(0,3) \boxplus (1,3)$.

(b) Determine $(0,3) \boxplus (1,-3)$.

(c) Determine $4(0,3)$, which is short for $(0,3) \boxplus (0,3) \boxplus (0,3) \boxplus (0,3)$.

Solution. We let Sage do the work for us:

```
>>> E = EllipticCurve([-1,9])
```

```
>>> E(0,3) + E(1,3)
```

$(-1:-3:1)$

```
>>> E(0,3) + E(1,-3)
```

$(35:207:1)$

```
>>> 4*E(0,3)
```

$$\left(-\frac{1677023}{60279696}:\frac{1406201395535}{468011559744}:1\right)$$

We conclude that $(0,3)\boxplus(1,3)=(-1,-3)$ and $(0,3)\boxplus(1,-3)=(35,207)$ (one of the points mentioned in Example 215), while

$$4(0,3)=\left(-\frac{1677023}{60279696},\frac{1406201395535}{468011559744}\right).$$

Comment. Note how Sage represents the point $(x,\,y)$ as $(x:\,y:\,1)$. These are **projective coordinates** which make it easier to incorporate the special point O which is represented by $(0:1:0)$.

https://en.wikipedia.org/wiki/Projective_coordinates

The following computation demonstrates that adding O doesn't do anything:

```
>>> E(0)
```

$(0:1:0)$

```
>>> E(0,3) + E(0)
```

$(0:3:1)$

Comment. Note that, starting with a single point such as $(0,\,-3)$, we can generate other points such as $2(0,-3)=\left(\frac{1}{36},\frac{647}{216}\right)$ (one of the points mentioned in Example 215). If the initial point is rational then so are the points generated from it.

Advanced comment. If you want to dig deeper, you can try to translate the geometric description of the addition $P\boxplus Q$ into algebra by deriving equations for the coordinates of $P\boxplus Q=(x_r,y_r)$ in terms of the coordinates of $P=(x_p,y_p)$ and $Q=(x_q,y_q)$. For instance, for the elliptic curve $y^2=x^3+ax+b$, one finds that

$$\begin{aligned} x_r &= \lambda^2-x_p-x_q,\\ y_r &= \lambda(x_p-x_r)-y_p, \end{aligned}$$

where $\lambda=(y_q-y_p)/(x_q-x_p)$ is the slope of the line connecting P and Q. If P and Q are the same point, then this line becomes the tangent line and the slope becomes $\lambda=(3x_p^2+a)/(2y_p)$ instead. For more details:

https://en.wikipedia.org/wiki/Elliptic_curve

From these formulas, can you reproduce the computations we did in Sage?

Review. Addition $P \boxplus Q$ of points P, Q on an elliptic curve.

For cryptographic purposes, elliptic curves are usually considered modulo a (large) prime p.

Example 218. Let us consider $y^2 = x^3 - x + 9$ (the elliptic curve from the previous examples) modulo 7. List all points on that curve.

Solution. Note that, because we are working modulo 7, there are only 7 possible values for each of x and y. Hence, we can just go through all $7^2 = 49$ possible points (x, y) to find all points on the curve.

Or, better, we go through all possibilities for x (such as $x = 2$) and determine the corresponding possible values for y (if $x = 2$, then $y^2 = 2^3 - 2 + 9 = 15 \equiv 1 \pmod 7$ which has solutions $y \equiv \pm 1 \pmod 7$).

Doing so, we find 9 points: $O, (0, \pm 3), (\pm 1, \pm 3), (2, \pm 1)$.

[Recall that O is the special point "at ∞" which serves as the neutral element with respect to $\boxplus$.]

Comment. A theorem of Hasse–Weil says that the number of points on an elliptic curve modulo p is always close to p (this is indeed what we expect because, for each of the p choices for x, we get an equation of the form $y^2 \equiv a \pmod p$ which has 2 solutions if a is a nonzero quadratic residue [and for a random a the odds are about 50% that it is quadratic]). Moreover, we can compute the exact number of points very efficiently.

By taking everything modulo 7, we still have the previously introduced addition rule $\boxplus$.

For instance. $(0, 3) \boxplus (1, -3) = (35, 207) \equiv (0, -3) \pmod 7$

Here is how we can use Sage to list all points, as well as add any two of them:

```
>>> E7 = EllipticCurve(GF(7), [-1,9])
>>> E7.points()
```

$[(0:1:0), (0:3:1), (0:4:1), (1:3:1), (1:4:1), (2:1:1), (2:6:1), (6:3:1), (6:4:1)]$

```
>>> E7(0,3) + E7(1,-3)
```

$(0:4:1)$

```
>>> E7(1,-3) + E7(0,-3)
```

$(6:3:1)$

Multiples of a point are simply denoted with nP. For instance, $3P = P \boxplus P \boxplus P$.

We then have a version of the **discrete logarithm** problem for elliptic curves:

(discrete logarithm) Given P, xP on an elliptic curve, determine x.
(computational Diffie–Hellman) Given P, xP, yP on an elliptic curve, determine $(xy)P$.

Comment. Interestingly, it appears that the computational Diffie–Hellman problem (CDH) is more difficult for elliptic curves modulo p than for regular multiplication modulo p. Indeed, suppose that p is an n-digit prime. Then the best known algorithms for regular CDH modulo p has runtime $2^{O(\sqrt[3]{n})}$, whereas the best algorithm for the elliptic curve CDH modulo p has runtime $\sqrt{p} \approx 2^{n/2} = 2^{O(n)}$.

As a consequence, it is believed that a smaller prime p can be used to achieve the same level of security when using elliptic curve Diffie–Hellman (ECDH). In practice 256bit primes are used, which is believed to provide security comparable to 2048bit regular Diffie–Hellman (DH); this makes ECDH about ten times faster in practice than DH.

Comment. On the other hand, due to that reduced bit size, quantum computing attacks on elliptic curve cryptography, if they become available, would be more feasible compared to attacks on ElGamal/RSA.

Armin Straub
straub@southalabama.edu

Comment. Apparently, more than 90% of webservers use one specific, NIST specified, elliptic curve: P-256:

$$y^2 = x^3 - 3x + 41058363725152142129326129780047268409114441015993725554835256314039467401291,$$

taken modulo $p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$ (the fact that $p \approx 2^{256}$ makes the computations on the elliptic curve much faster in practice). The initial point $P = (x, y)$ on the curve has huge coordinates as well.

Using this single curve is sometimes considered to be problematic, especially following the concerns that the NSA may have implemented a backdoor into Dual_EC_DRBG, which was a NIST standard 2006–2014.

https://en.wikipedia.org/wiki/Dual_EC_DRBG

A popular alternative is the curve Curve25519. In addition to some desirable theoretical advantages, its parameters are small and therefore not of similarly mysterious origin as the ones for P-256:

$$y^2 = x^3 + 486662x^2 + x, \quad p = 2^{255} - 19, \quad x = 9.$$

[Instead of points with (x, y) coordinates, one can actually work with just the x-coordinates for an additional speed-up.]

https://en.wikipedia.org/wiki/Curve25519

```
>>> E = EllipticCurve(GF(2^255-19), [0,486662,0,1,0])
```

```
>>> E
```

$y^2 = x^3 + 486662\,x^2 + x$

```
>>> E.order()
```

57896044618658097711785492504343953926856930875039260848015607506283634007912

```
>>> log(E.order(),2).n()
```

255.000000000000

```
>>> P = E.lift_x(9)
```

```
>>> P
```

(9: 43114425171068552920764898935933967039370386198203806730763910166200978582548: 1)

```
>>> 100*P
```

(44032819295671302737126221960004779200206561247519912509082330344845040669336: 8626006\
39244757237163427806001665957575078127166632317389150490196167274334 4: 1)

```
>>> P.order()
```

7237005577332262213973186563042994240857116359379907606001950938285454250989

```
>>> log(P.order(),2).n()
```

252.000000000000

```
>>> E.order() / P.order()
```

8

```
>>> 5*(20*P) == 20*(5*P)
```

1

On the other hand, it should be pointed out that it is not an easy task to "randomly generate" cryptographically secure elliptic curves plus suitable base point. That's the reason pre-selected elliptic curves are of importance.

http://blog.bjrn.se/2015/07/lets-construct-elliptic-curve.html

P versus NP: A Millennium Prize Problem

The Clay Mathematics Institute has offered 10^6 dollars each for the first correct solution to seven **Millennium Prize Problems**. Six of the seven problems remain open.

https://en.wikipedia.org/wiki/Millennium_Prize_Problems

Comment. Grigori Perelman solved the Poincaré conjecture in 2003 (but refused the prize money in 2010).
https://en.wikipedia.org/wiki/Poincaré_conjecture

Example 219. (P vs NP) P versus NP is one of the Millennium Prize Problems that is of particular importance to cryptography.

"*If the solution to a problem is easy to check for correctness, is the problem easy to solve?*"

https://en.wikipedia.org/wiki/P_versus_NP_problem

Roughly speaking, consider decision problems which have an answer of yes or no. P is the class of such problems, which can be solved efficiently. NP are those problems, for which we can quickly verify that the answer is yes if presented with suitable evidence.

For instance.

- It is unknown whether factoring (in the sense of: does N have a factor $\leqslant M$?) belongs to P or not.
  The problem is definitely in NP because, if presented with a factor $\leqslant M$, we can easily check that.
- Deciding primality is in P (maybe not so shocking since there are very efficient nondeterministic algorithms for checking primality; not so for factoring).
- In the (decisional) travelling salesman problem, given a list of cities, their distances and d, the task is to decide whether a route of length at most d exists, which visits each city exactly once.
  The decisional TSP is clearly in NP (take as evidence the route of length $\leqslant d$). In fact, the problem is known to be NP-complete, meaning that it is in NP and as "hard" as possible (in the sense that if it actually is in P, then $\mathrm{P}{=}\mathrm{NP}$; that is, we can solve any other problem in NP efficiently).
- Other NP-complete problems include:
  - Sudoku: Does a partially filled grid have a legal solution?
  - Subset sum problem: Given a finite set of integers, is there a non-empty subset that sums to 0?

Comment. "Efficiently" means that the problem can be solved in time polynomial in the input size.
Take for instance computing $2^n \pmod n$, where n is the input (it has size $\log_2(n)$). This can be done in polynomial time if we use binary exponentiation (whereas the naive approach takes time exponential in $\log_2(n)$).

Comment. This is one of the few prominent mathematical problems which doesn't have a definite consensus. For instance, in a 2012 poll of 151 researchers, about 85% believed $\mathrm{P}{\neq}\mathrm{NP}$ while about 10% believed $\mathrm{P}{=}\mathrm{NP}$.

Comment. NP are problems that can be verified efficiently if the answer is "yes". Similarly, $\mathrm{co\text{-}NP}$ are problems that can be verified efficiently if the answer is "no". It is an open problem whether $\mathrm{NP}{\neq}\mathrm{co\text{-}NP}$.

- Factoring is in both NP and $\mathrm{co\text{-}NP}$ (it is in $\mathrm{co\text{-}NP}$ because primality testing is in P).
- For all NP-complete problems it is unknown whether they are in $\mathrm{co\text{-}NP}$. (If one of them is, then we would, unexpectedly, have $\mathrm{NP}{=}\mathrm{co\text{-}NP}$.)

Armin Straub
straub@southalabama.edu

The Riemann hypothesis: Another Millennium Prize Problem

The Riemann hypothesis is another one of the seven Millennium Prize Problems that is of importance to the underpinnings of cryptography. It is concerned with the distribution of primes.

Recall that we discussed the prime number theorem, which states that, up to x, there are about $x/\ln(x)$ many primes. The Riemann hypothesis gives very precise error estimates for an improved prime number theorem (using a function more complicated than the logarithm).

Example 220. (Riemann hypothesis) Consider the **Riemann zeta function** $\zeta(s) = \sum_{n \geqslant 1} \frac{1}{n^s}$. This series converges (for real s) if and only if $s > 1$.

The divergent series $\zeta(1)$ is the harmonic series, and $\zeta(p)$ is often called a p-series in Calculus II.

Comment. Euler achieved worldwide fame in 1734 by discovering and proving that $\zeta(2) = \frac{\pi^2}{6}$ (and similar formulas for $\zeta(4), \zeta(6), \ldots$).

For complex values of $s \neq 1$, there is a unique way to "analytically continue" this function. It is then "easy" to see that $\zeta(-2) = 0$, $\zeta(-4) = 0$, ... The **Riemann hypothesis** claims that all other zeroes of $\zeta(s)$ lie on the line $s = \frac{1}{2} + a\sqrt{-1}$ ($a \in \mathbb{R}$). A proof of this conjecture (checked for the first 10,000,000,000 zeroes) is worth $1,000,000.

http://www.claymath.org/millennium-problems/riemann-hypothesis

The connection to primes. Here's a vague indication that $\zeta(s)$ is intimately connected to prime numbers:

$$\begin{aligned} \zeta(s) &= \left(1 + \frac{1}{2^s} + \frac{1}{2^{2s}} + \ldots\right)\left(1 + \frac{1}{3^s} + \frac{1}{3^{2s}} + \ldots\right)\left(1 + \frac{1}{5^s} + \frac{1}{5^{2s}} + \ldots\right)\cdots \\ &= \frac{1}{1-2^{-s}} \frac{1}{1-3^{-s}} \frac{1}{1-5^{-s}} \cdots \\ &= \prod_{p \text{ prime}} \frac{1}{1-p^{-s}} \end{aligned}$$

This infinite product is called the Euler product for the zeta function. If the Riemann hypothesis was true, then we would be better able to estimate the number $\pi(x)$ of primes $p \leqslant x$.

More generally, certain statements about the zeta function can be translated to statements about primes. For instance, the (non-obvious!) fact that $\zeta(s)$ has no zeros for $\operatorname{Re} s = 1$ implies the prime number theorem.

http://www-users.math.umn.edu/~garrett/m/v/pnt.pdf