Lecture 1: Random numbers, coin tossing

This lecture and the next are part of the CryptoWorks21 Novice program in quantum cryptography tools. The third and fourth lectures are not part of the the CryptoWorks 21 Novice program.

1 Random number generation

1.1 In a classical universe

True randomness is impossible in a classical universe. If you know enough about the present then you can predict the future and the past.

Many important tasks require randomness. (e.g. games and gambling, algorithms, cryptograhy.) Thus, many ad hoc random number generators have been used over the years.

Mechanical methods. Dice, coin flipping, roulette wheel, card shuffling.
Advantages: Very good randomness. Entertaining.
Disadvantages: Too slow for anything except entertainment.

For the remainder of this section we focus on...

Algorithmic methods. You must start with at least some small random *seed*, otherwise you're pooched—an adversary can repeat your computation and hence predict your "randomness".

Pseudorandom number generators try to "stretch" a random seed into a larger random string.

Definition 1 (Pseudorandom generator (PRG))**.** A function $R : \{\text{small strings}\} \to \{\text{large strings}\}$ is a *pseudorandom generator* if, given uniformly random bits as input (the *seed*), no efficient procedure can reliably distinguish the output of R from truly random bits. □

Everyone believes that PRGs exist and there are many candidate PRGs. However, PRGs are known to exist only under unproven (but widely believed) complexity-theoretic assumptions such as the existence of one-way functions. It would be nice if PRGs could be constructed by assuming only $\mathrm{P} \neq \mathrm{NP}$ or, failing that, that NP-complete problems are hard on average. But nobody knows how to do that.

Advantages: Very fast. Good enough for many purposes.
Disadvantages: Trade-off between speed and quality of randomness. Very little in the way of theoretical guarantees. Still need a random seed.

The linear congruential generator. This is the most commonly used PRG. Parameterized by integers m (the *modulus*) and $a, c \in \mathbb{Z}_m$. Given a seed value $x_0 \in \mathbb{Z}_m$, subsequent "random" numbers $x_1, x_2, \ldots$ are given by

$$x_{i+1} \equiv a x_i + c \pmod{m}.$$

Seems ridiculous, doesn't it? Many popular programming languages use this PRG!
e.g. Microsoft Visual C++ uses $m = 2^{32}$, $a = 214013$, $c = 2531011$, and returns bits $30 \ldots 16$ of x_i.

How to get the random seed? Use a slow-but-good random number generator. Poll the current time-of-day. (e.g. millisecond digit is assumed to be random.)

This is not just a mathematical curiosity! Generating good randomness is a real problem in today's world!

Urban legend. Some math whiz studies a video keno machine at a casino. Eventually, she breaks the PRG and predicts the next game's numbers and wins a million dollars. (I think something like this actually happened in real life.)

Cool anecdote. RSA is currently the dominant public-key encryption scheme for internet traffic. An RSA public-key contains a large integer $N = pq$, where p, q are large primes that are secret (the private key). If you can factor large integers then you can break RSA. Shor's algorithm breaks RSA.

RSA keys are supposed to be constructed from random primes p, q. By gathering a large database of 12 million RSA keys, in 2012 Lenstra *et al.* [LHA$^+$12] were able to crack 2 out of every 1000 RSA keys via a simple pairwise GCD algorithm! Something similar was done independently by Heninger *et al.* at the same time [HDWH12].

If all the keys were generated from truly random primes p, q then the probability of cracking so many keys is vanishingly small. Far too many keys shared a prime factor with other keys. Someone wasn't using a good enough PRG! This story even made the *New York Times*.

1.2 In a quantum universe

True randomness is easy! Just measure a $|+\rangle$ state in the $\{|0\rangle, |1\rangle\}$ basis. We can all go home now.

Wait! We need a harder problem to justify our grants. Okay, consider this: company R-Wave sells magic boxes that it claims produce truly random bits on demand. Can you trust R-Wave's device? Is there a way to verify that you are receiving true randomness?

This question is an example of a question about *device-independent cryptography*: is there a protocol that produces certifiably random bits, even if the other parties (devices) are cheating?

The problem of device-independent randomness was recently solved independently by three groups: [FGS13, PM13, VV12]. I will present Vazirani-Vidick here.

What they accomplish: Given a security parameter $k \in \mathbb{N}$ and nk random "seed" bits, use a completely untrusted no-signaling box to produce $N = 2^n$ bits that are "2^{-k}-close" to truly random.

Observe:

1. We still require a random seed, so the protocol *stretches* a small amount of randomness into a large amount.

2. We make no assumptions about the box other than that it obeys quantum mechanics and that A, B cannot communicate.

Compare:

- *Classical PRGs...* ...use a *trusted* device to achieve only *polynomial* stretch that only *appears* random to a computationally-bounded observer (we hope).

- *Quantum VV protocol...* ...uses an *untrusted* device to achieve *exponential* stretch that is guaranteed to be *truly* random.

Main idea: Play a repeated CHSH game (see Figure 1) with the untrusted box on mostly zero-inputs. Throw in a few "blocks" of repetitions with truly random inputs to keep the box honest. If the box can with the CHSH game $\approx$85% of the time within each block then it must be playing $\approx$ honestly on *every* repetition. In this case, the bits produced by the box must be $\approx$ random.

Is device-independent quantum randomness practical? Not yet. Will it ever be? Who knows?

2 Coin tossing

Coin tossing is a cryptographic primitive introduced in 1981 by Blum [Blu81]. A protocol for coin tossing allows two parties who do not trust each other to agree on a uniformly random bit. In the words of Blum:

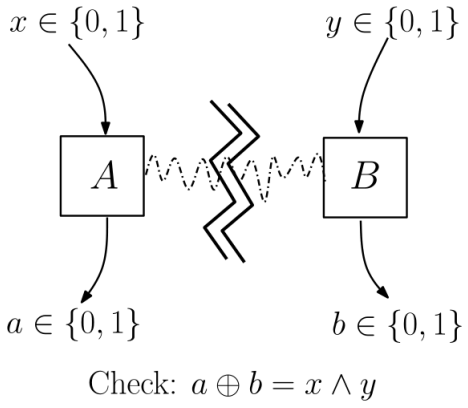

Figure 1: The famous CHSH game. Classical players can win with probability no larger than 75%. Quantum players who share EPR pairs can win with probability ≈85%.

> Alice and Bob want to flip a coin by telephone. (They have just divorced, live in different cities, want to decide who gets the car.) ... Alice (at the other end of the line) says, "Here goes... I'm flipping the coin... You lost!"

We distinguish between two types of coin tossing primitives.

- *Strong coin tossing:* A cheating party cannot bias the honest party's outcome toward either 0 or 1.
- *Weak coin tossing:* Each party has a preferred outcome and we stipulate only that a cheating party cannot bias the output toward his or her preferred outcome.

Here's a formal definition.

Definition 2 (Strong, weak coin tossing)**.** A protocol for *strong coin tossing with bias* $\varepsilon \geq 0$ must meet the following conditions:

1. If both parties are honest then they both produce a bit $c \in \{0,1\}$ generated uniformly at random.
2. For each $c \in \{0,1\}$, no honest party can be forced to produce c with probably greater than $1/2 + \varepsilon$.

For *weak coin tossing*, item 2 above is replaced with

2(a). If Alice is honest then she cannot be forced to produce 1 with probability greater than $1/2 + \varepsilon$.

2(b). If Bob is honest then he cannot be forced to produce 0 with probability greater than $1/2 + \varepsilon$.

Of course, each party is allowed a local source of true randomness. The problem is that neither party trusts the other to make honest use of that randomness. □

By definition, strong coin tossing is at least as hard as weak coin tossing.

2.1 In a classical universe

Blum proved that strong coin tossing with arbitrarily small bias can be achieved assuming the existence of one-way functions. (If you didn't already know, a *one-way function* is a function that is easy to compute but hard to invert.) He provides an explicit protocol for strong coin tossing based on the hardness of factoring.

What can be said about coin tossing in a classical universe without any computational assumptions? It turns out that even weak coin tossing is completely impossible in this setting.

Proposition 3 (Impossibility of classical coin tossing)**.** *In the absence of computational assumptions, for each classical coin tossing protocol that meets condition 1 for honest parties there exists a cheating party that can force any desired outcome upon the honest party with certainty.*

Thus, the question of classical coin tossing is pretty much open-and-shut, thanks to Blum. In particular:

1. If computational assumptions are allowed then strong coin tossing can be achieved with arbitrarily small bias.

2. If computational assumptions are not allowed then even weak coin tossing with bias less than $1/2$ is impossible.

Well, okay, it's not *completely* open-and-shut. Here are a couple questions that seem to be unresolved:

1. Can the computational assumptions required to achieve strong coin tossing be weakened? Factoring is easy on a quantum computer; does there exist a quantum-safe coin tossing protocol based on, say, lattices or something?

2. What about trade-offs between conditions 1 and 2 in Definition 2? *e.g.* If we relax the requirement that honest parties *always* agree then does coin tossing become possible?

2.2 In a quantum universe

Coin tossing is totally impossible without computational assumptions in a classical universe; can quantum information help? Yes, it can!

The bad news:

- *Strong coin tossing:* Kitaev proved that any protocol must have bias at least $1/\sqrt{2} - 1/2 \approx 0.207$ [Kit02]. That is, for some $c \in \{0, 1\}$ a cheating party can always force an honest party to produce c with probability at least $1/\sqrt{2} \approx 0.707$.

  (This result was a homework question in a module I taught for last year's offering of this course.)

- *Weak coin tossing:* Ambainis proved that any protocol for weak coin tossing that achieves bias ε must involve at least $\Omega\left(\log\log(1/\varepsilon)\right)$ messages [Amb04].

The good news:

- *Strong coin tossing:*

  1. Ambainis and Spekkens-Rudolph independently discovered protocols that achieve bias $1/4$, which is only slightly worse than the best possible 0.207 [Amb04, SR02].
  2. Chailloux and Kerenidis give a procedure that turns any weak coin tossing protocol with bias ε into a strong coin tossing protocol with bias $1/\sqrt{2} - 1/2 + O(\varepsilon)$ [CK09]. Thus, arbitrarily good weak coin tossing protocols imply arbitrarily close-to-optimal strong coin tossing protocols.

- *Weak coin tossing:*

  1. Spekkens-Rudolph gave a protocol that achieves bias $1/\sqrt{2} - 1/2$, which is the best possible for *strong* coin tossing.
  2. Mochon made it a personal mission to solve this problem. He presented a series of papers that surpass the best possible bias for strong coin tossing: 0.192, then $1/6$. Finally, building on unpublished work of Kitaev, Mochon the limit [Moc07]; he found a protocol that achieves bias ε for any desired $\varepsilon > 0$.

     (It then follows from Chailloux-Kerenidis that there is also an optimal protocol for *strong* coin tossing, too.)

Legend. Mochon posted his tour-de-force to arxiv and then got a job in the finance industry. No one understands Mochon's paper and Mochon has disappeared. Chailloux and others are attempting to decypher it. I've looked into it a bit and got nowhere.

References

[Amb04] Andris Ambainis. A new protocol and lower bounds for quantum coin flipping. *Journal of Computer and System Sciences*, 68(2):398–416, 2004. arXiv:quant-ph/0204022. 4

[Blu81] Manuel Blum. Coin flipping by telephone. In *Advances in Cryptology—CRYPTO '81*, pages 11–15, 1981. 2

[CK09] André Chailloux and Iordanis Kerenidis. Optimal quantum strong coin flipping. In *Proceedings of the 50th IEEE Symposium on Foundations of Computer Science*, FOCS '09, pages 527–533, 2009. arXiv:0904.1511 [quant-ph]. 4

[FGS13] Serge Fehr, Ran Gelles, and Christian Schaffner. Security and composability of randomness expansion from Bell inequalities. *Physical Review A*, 87(1):012335, 2013. arXiv:1111.6052 [quant-ph]. 2

[HDWH12] Nadia Heninger, Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. Mining your Ps and Qs: Detection of widespread weak keys in network devices. In *Proceedings of the 21st USENIX Security Symposium*, 2012. 2

[Kit02] Alexei Kitaev. Quantum coin-flipping. Presentation at the 6th Workshop on *Quantum Information Processing* (QIP 2003), 2002. 4

[LHA+12] Arjen Lenstra, James Hughes, Maxime Augier, Joppe Bos, Thorsten Kleinjung, and Christophe Wachter. Ron was wrong, Whit is right. Cryptology ePrint Archive, Report 2012/064, 2012. 2

[Moc07] Carlos Mochon. Quantum weak coin-flipping with arbitrarily small bias, 2007. arXiv:0711.4114 [quant-ph]. 5

[PM13] Stefano Pironio and Serge Massar. Security of practical private randomness generation. *Physical Review A*, 87(1):012336, 2013. arXiv:1111.6056 [quant-ph]. 2

[SR02] R. Spekkens and T. Rudolph. Degrees of concealment and bindingness in quantum bit-commitment protocols. *Physical Review A*, 65: article 123410, 2002. arXiv:quant-ph/0106019v2. 4

[VV12] Umesh Vazirani and Thomas Vidick. Certifiable quantum dice: or, true random number generation secure against quantum adversaries. In *Proceedings of the 44th ACM Symposium on Theory of Computing (STOC 2012)*, pages 61–76, 2012. arXiv:1111.6054 [quant-ph]. 2