QRAND

A quantum random number generator for arbitrary probability distributions

Pedro Rivero – March 10, 2021

Unitary Fund

Table of contents

The need for *random numbers**

Applications

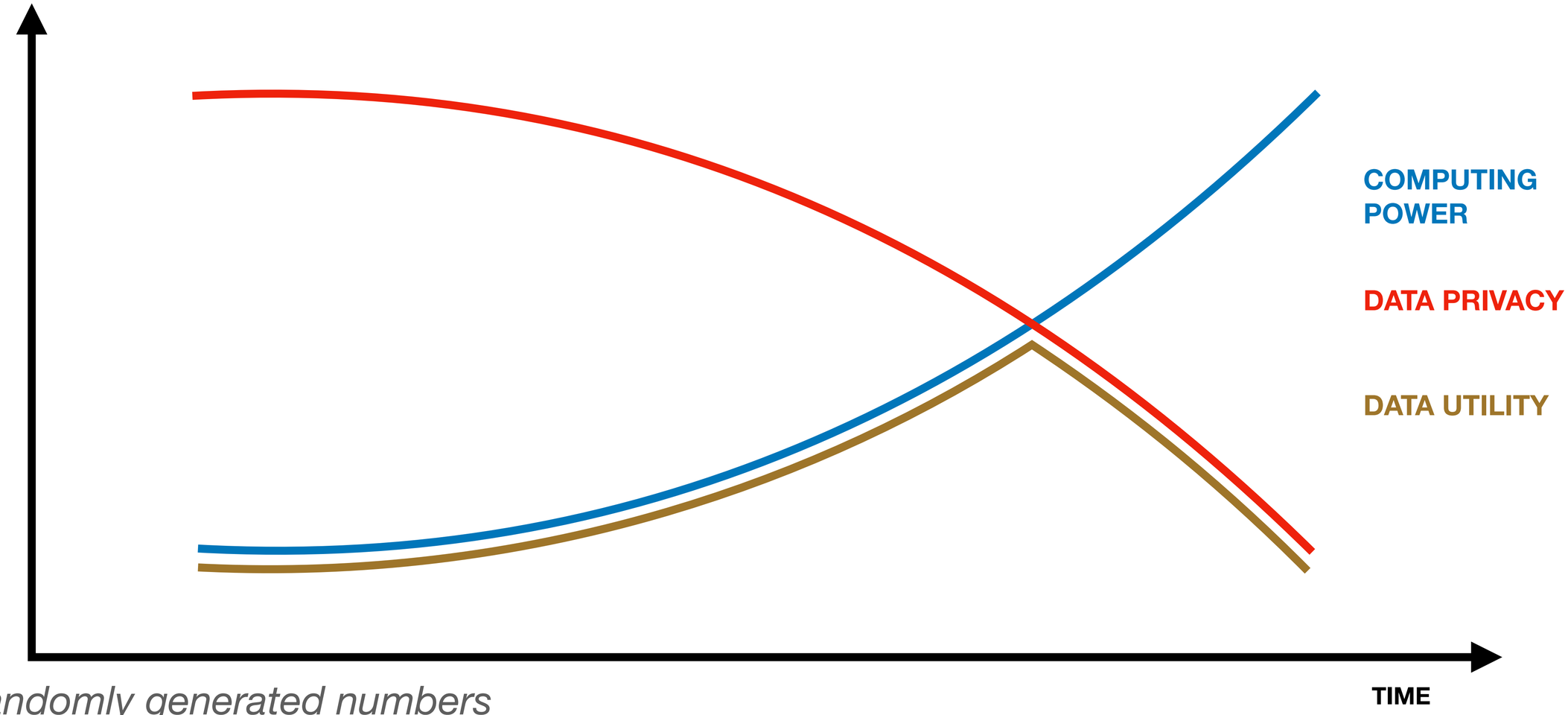
- Information/communications security (e.g. crypto-systems, IoT)
- Computation (e.g. algorithms, physical simulations, Monte Carlo)
- Financial systems (e.g. transactions, credit card chips)
- Statistical sampling (e.g. scientific experiments, social studies)
- Testing (e.g. randomized benchmarking)
- Legal and trust sensitive processes (e.g. jury, lotteries, *Kleroterion*)

**Randomly generated numbers*


cnr

The need for *random numbers**

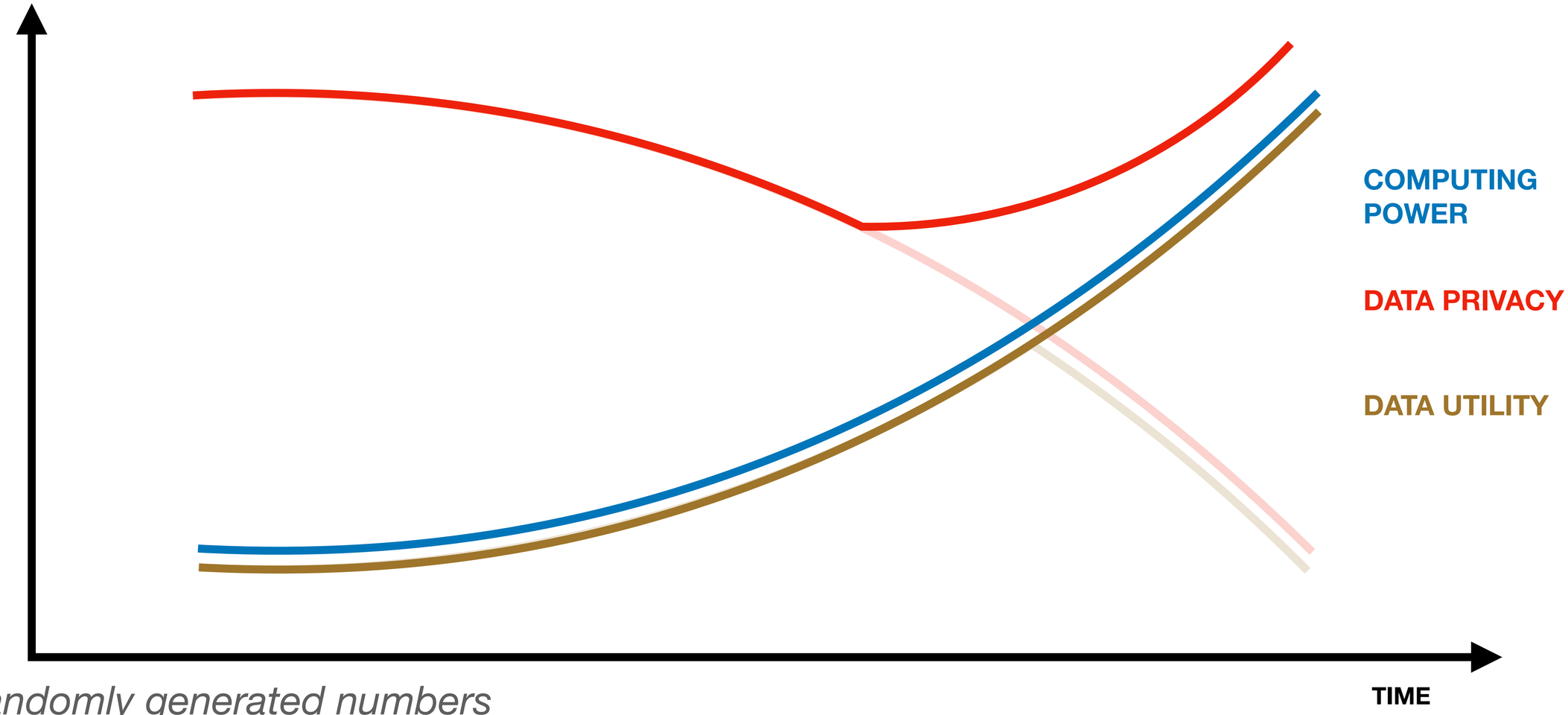
Data utility curve



*Randomly generated numbers

The need for *random numbers**

Data utility curve



*Randomly generated numbers

What we mean by random

A slippery concept

- The attribute of being random applies more correctly to a sequence of numbers.
- It is strictly related to lack/impossibility of predictability.
- The impossibility of compression can be considered as the defining trait of randomness (Chaitin & Kolmogorov).

- Random number generator → R(NG) ≠ (RN)G

Randomness quality

Entropy sources

- It is nearly impossible to establish whether a source (of bit strings) is actually random.
- We can analyze the general properties of output strings (e.g. weight)
- **Shannon Entropy** (H) → Measure of disinformation/uncertainty

> *The minimum number of yes/no questions that need to be answered about any output to be able to guess it correctly with 100% confidence.*

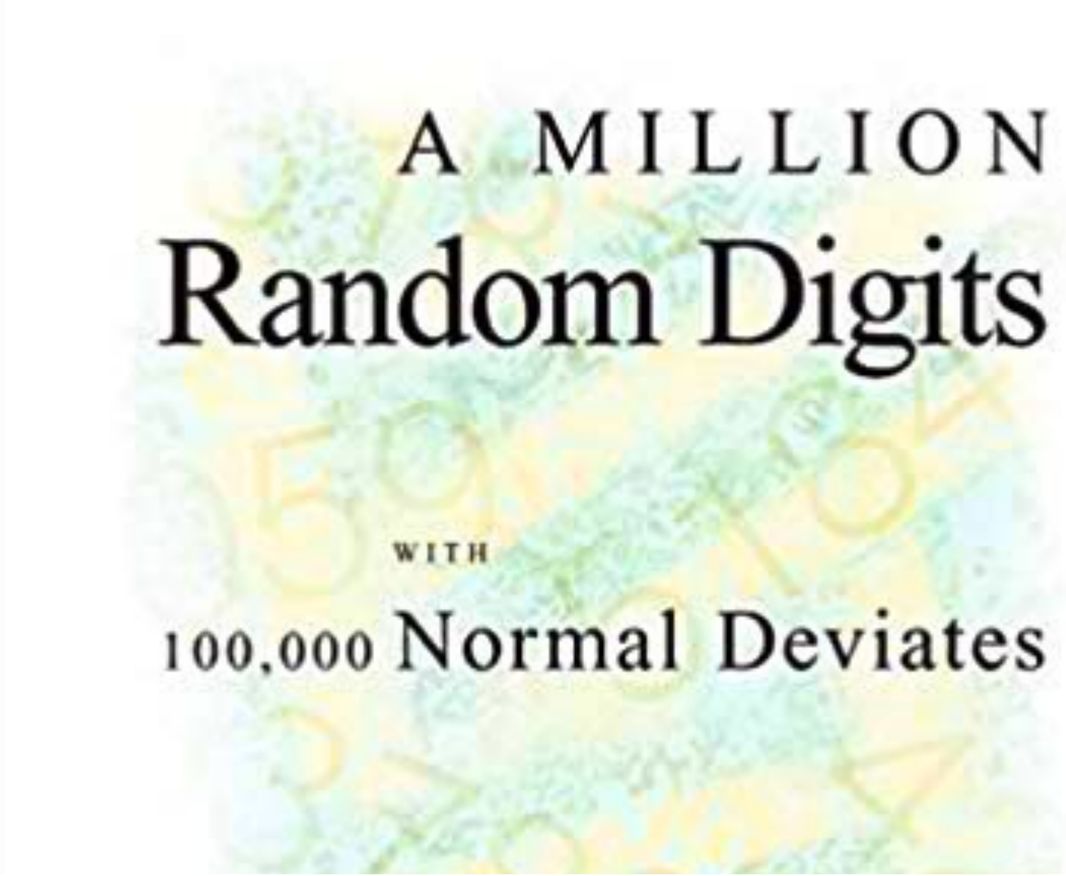
- Bit strings which are not random have limited entropy and can therefore be described with fewer bits than the string itself.

Exemplar bit string produced	String probability assuming 0 and 1 equally likely and bits are independent	Fraction of zeros	Displays Pattern?	Is the source random?
0 0 0 0 0 0 0 0 0 0 **H = 0**	2^{-10}	100%	Yes	It is reasonable to suspect it is not; it might well be that the source outputs only 0s
0 1 0 1 0 1 0 1 0 1 **H = 1**	2^{-10}	50%	Yes	It is reasonable to suspect it is not; it might well be that the source output simply switches between 0 and 1
0 0 1 0 0 0 0 0 0 1 $H = -\sum p(x) \log p(x)$ **H ≈ 7.2 < 10**	2^{-10}	80%	No	Potentially yes, but it appears but it appears to be biased; it may indicate that 0 and 1 are not equally likely – that is, the assumption of 0 and 1 being actually likely may not hold
1 0 1 1 0 1 0 0 1 0 **H = 10**	2^{-10}	50%	No	Potentially yes, and it appears also unbiased; 0 and 1 may really be equally likely

Randomness quality

Output tests

- There is no way of knowing if the source is truly random.
- A passing output will always pass (the tests).
- An adversary might *predict* or *fabricate* the output.
- Numbers are not inherently random *per se*.



TDB

★☆☆☆☆ **Not really random**

Reviewed in the United States on September 26, 2012

I bought two copies of this book. I find that the first copy perfectly predicts what the numbers will be in the second copy. I feel cheated.

141 people found this helpful

NIST Randomness Beacon

Not private!

Should we trust it?

Random number generation

Properties

- **Uniformity** → all sequences have the same probability of occurring
- **Scalability** → any test applicable to a sequence must must apply to subseq.
- **Consistency** →the behavior must be consistent across all outputs
- **Forward unpredictability** → the next output should not be predictable
- **Backward unpredictability** → previous outputs should not be predictable

Random number generation

The challenge

- Generating random numbers is hard → you need to get your hands dirty and start experimenting with physical hardware.

RANDOM PROCESS

To conveniently sample from for getting entropy

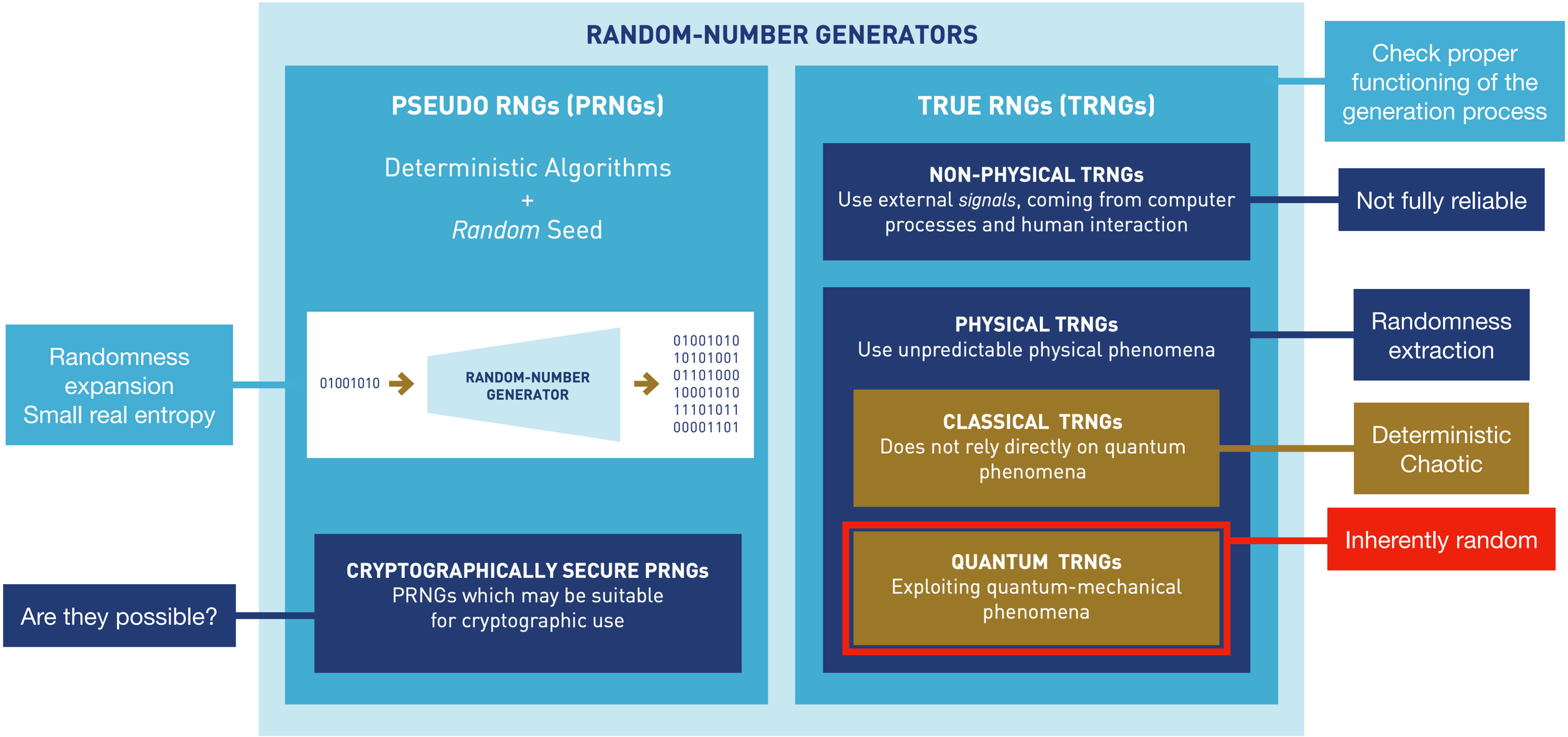
NO SECURITY BY OBSCURITY

Transparent entropy sources

RANDOM NUMBER DELIVERY

Keep the entropy pure (no leaks)

Random number generation



Random number generation

Classical vs Quantum

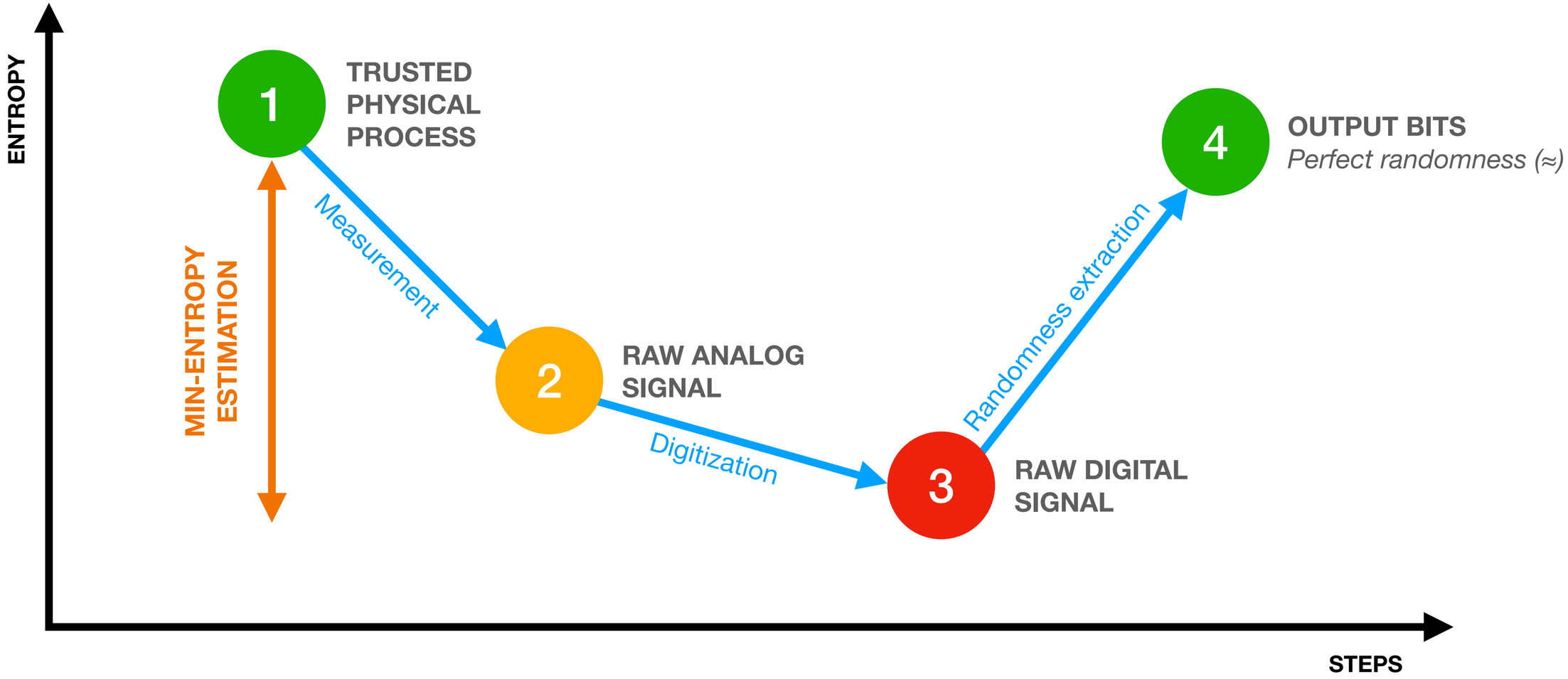
- All processes in classical physics are inherently deterministic.

> We may regard the present state of the universe as the effect of its past and the cause of its future. An intellect which at a certain moment would know all forces that set nature in motion, and all positions of all items of which nature is composed, if this intellect were also vast enough to submit these data to analysis, it would embrace in a single formula the movements of the greatest bodies of the universe and those of the tiniest atom; for such an intellect nothing would be uncertain and the future just like the past would be present before its eyes. — *Pierre-Simon Laplace*

- We need to resort to quantum physics to find truly physical randomness.

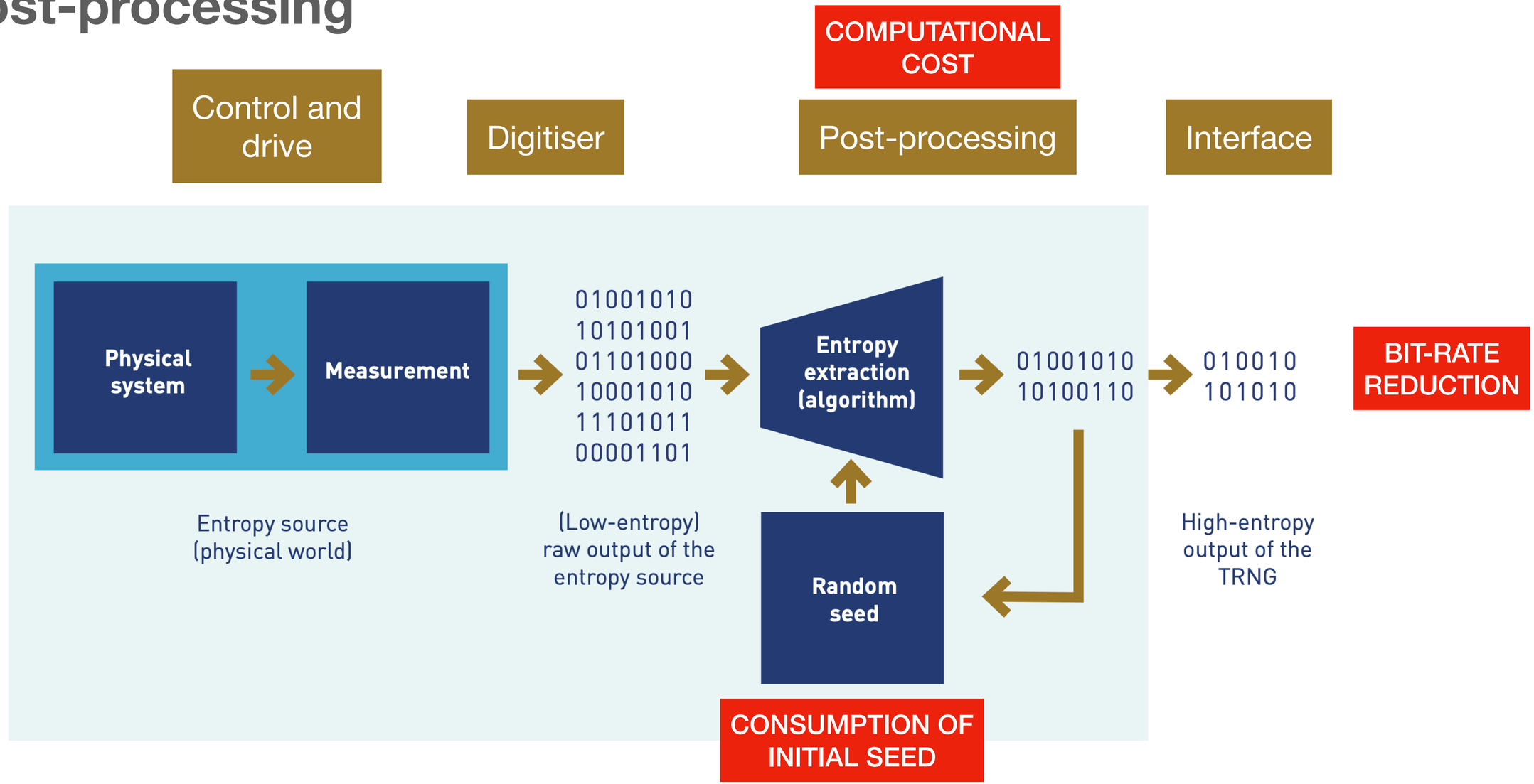
Randomness extraction

Post-processing: randomness metrology



Randomness extraction

Post-processing

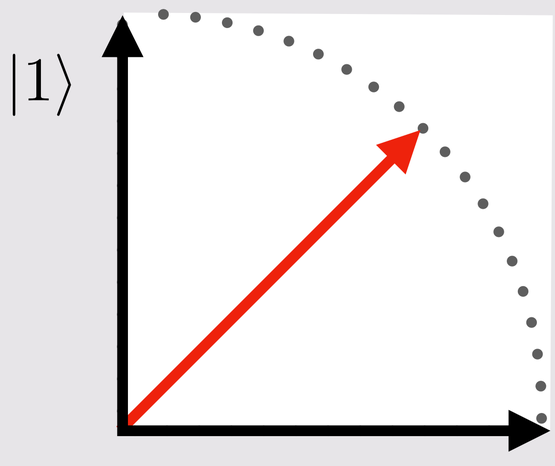


Quantum random number generators

Quantum mechanics primer

$$\text{Bits: } \{0, 1\} \quad \rightarrow \quad \text{Qubits: } \{|0\rangle, |1\rangle\}$$

- **Qubits** → quantum system with two possible (distinguishable) states
- **Superposition** → a quantum system can be in a state which is a mix of classically distinguishable states.
- **Inherently random measurements**
- **Entanglement** → classically inexplicable correlations between subsystems

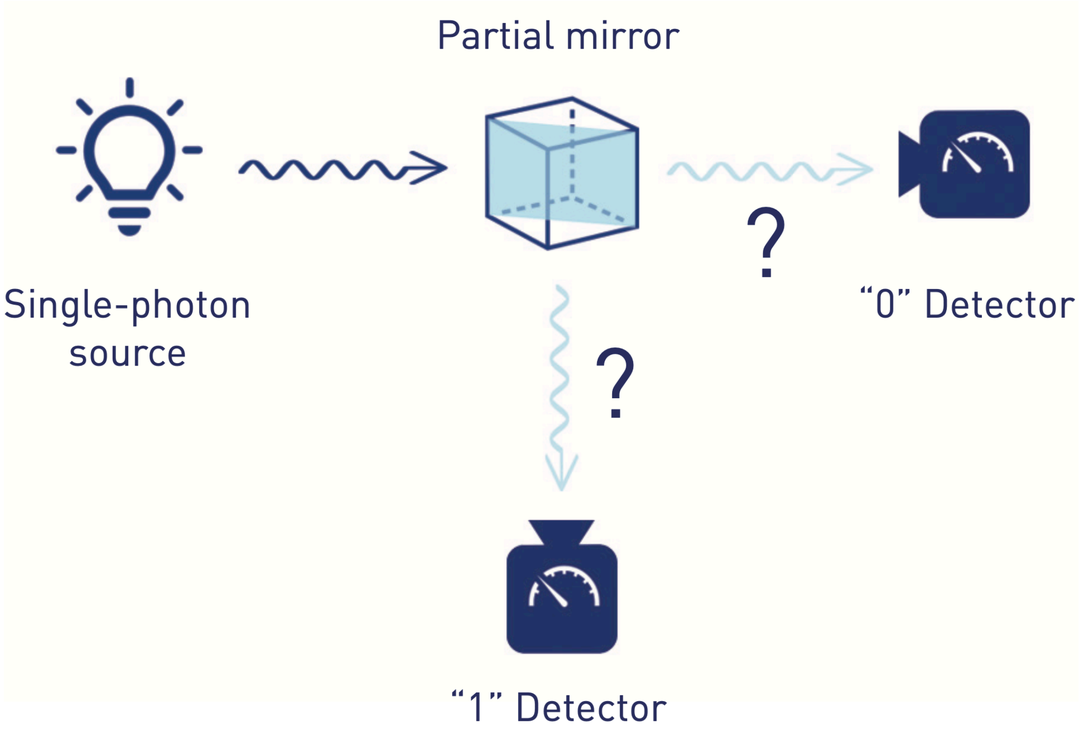


$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

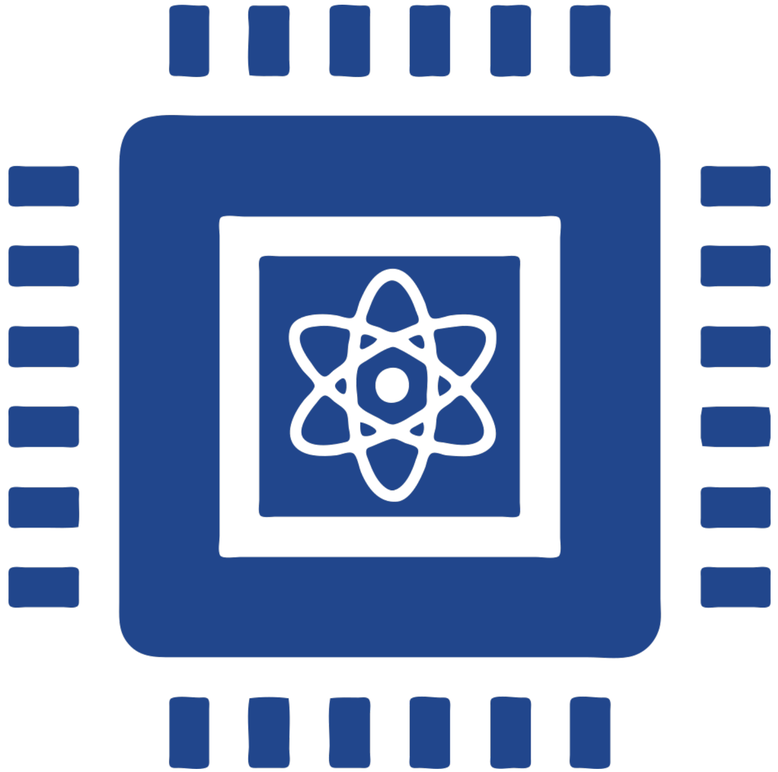
$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

Quantum random number generators

Practical realizations



Analog quantum hardware



Digital quantum computer

Quantum random number generators

Trade-offs of quantum RNGs

ENTROPY SOURCE

Fundamentally random

ENTROPY QUALITY

High from the start

CERTIFICATION

Can validate the underlying process

SECURITY

Built-in with device independent techniques

SPEED

High for its quality
Slower on digital quantum computers

SIZE

Variable, from very small to room size

VERSATILITY

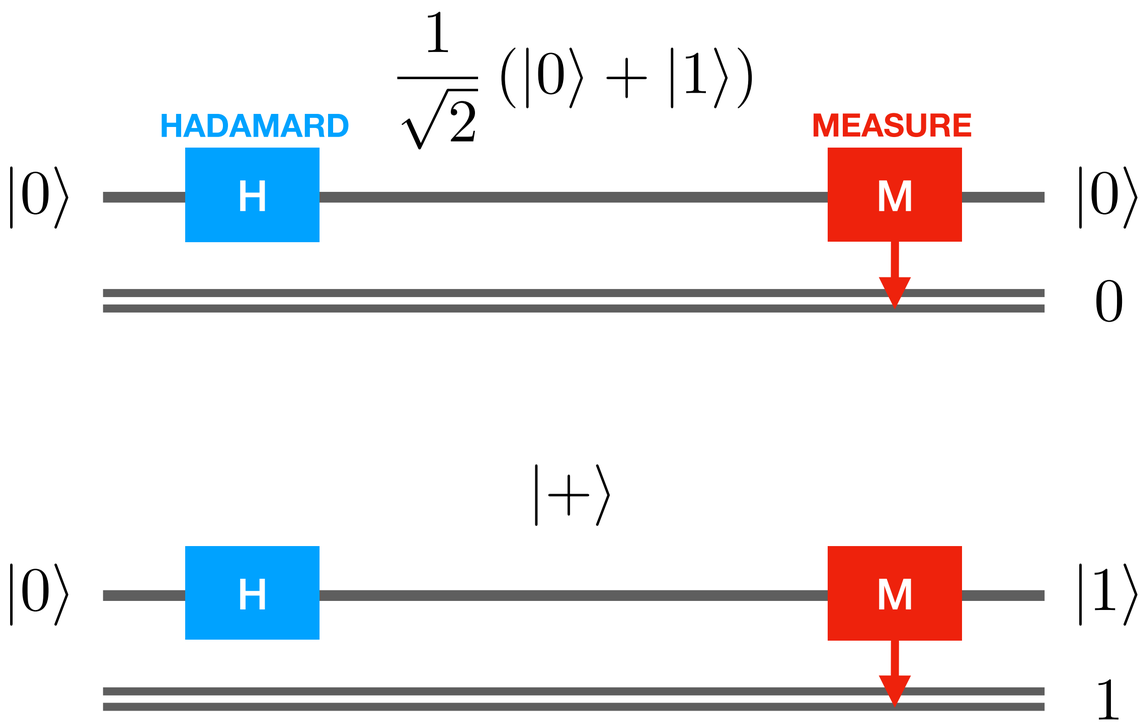
Easily scalable, specially with universal quantum computers

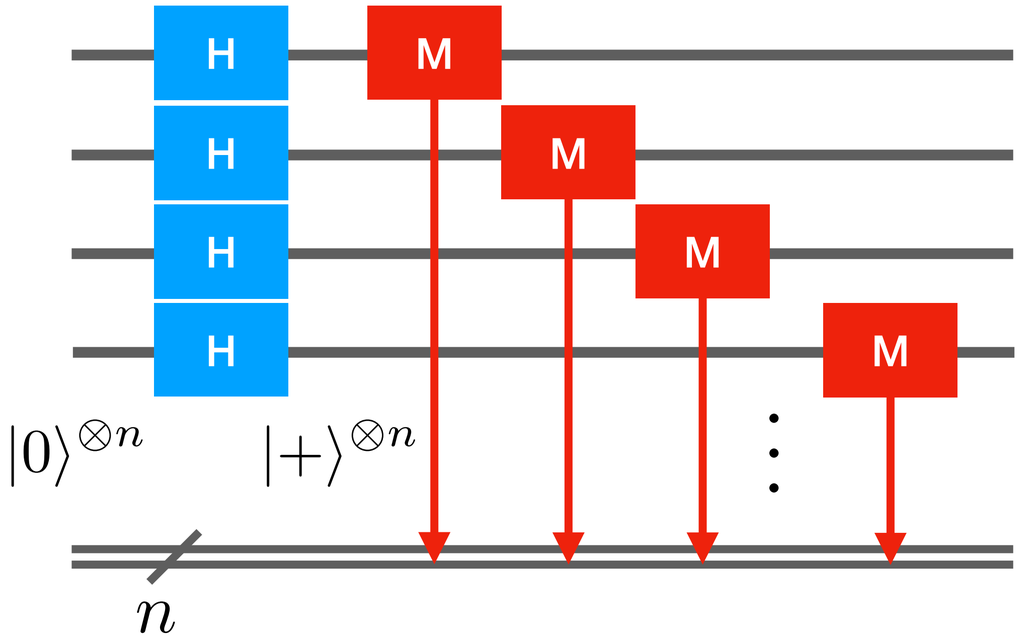
SUSTAINABILITY

Better for universal quantum computers

Quantum random number generators

The naive approach using quantum circuits





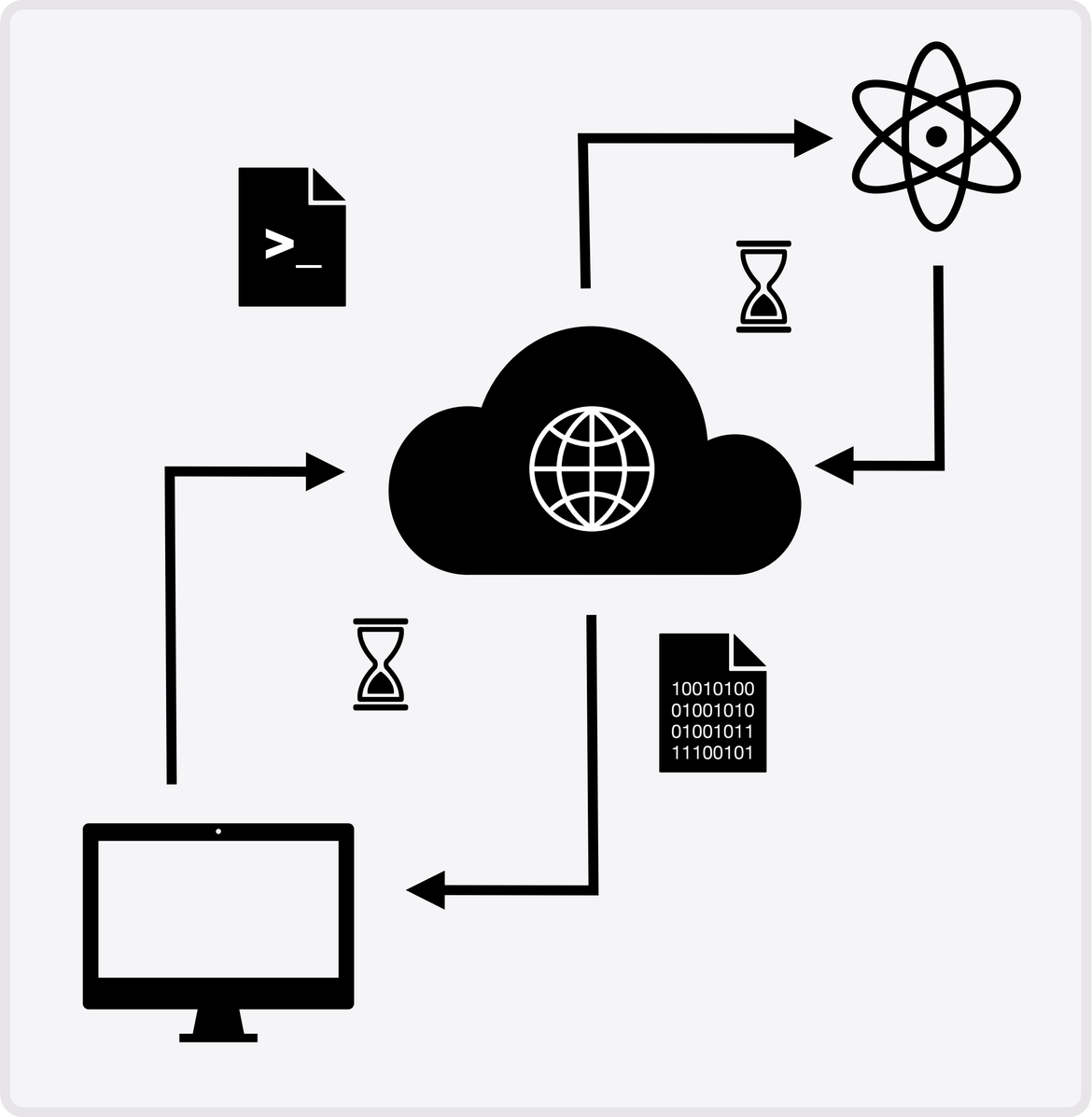
VIDEO

YouTube: Pedro Rivero QRAND

QRAND

A python tool (v0.2.0 PyPI)

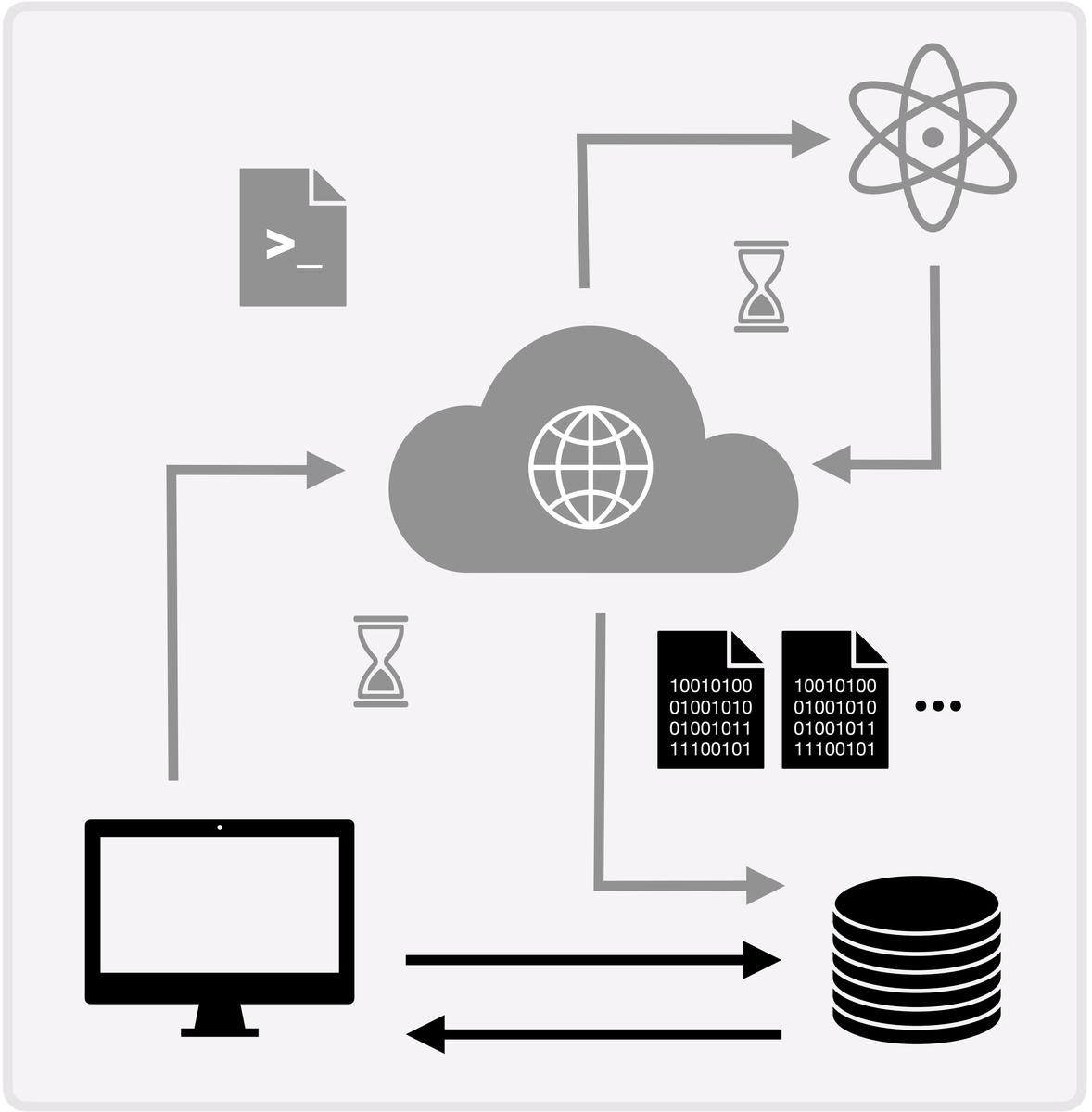
- Efficiency sampling
- API
- NumPy integration
- Extended functionality
- QRNG (PyPI) integration



QRAND

A python tool (v0.2.0 PyPI)

- **Efficiency sampling**
- API
- NumPy integration
- Extended functionality
- QRNG (PyPI) integration



QRAND

A python tool (v0.2.0 PyPI)

- Efficiency sampling
- **API**
- NumPy integration
- Extended functionality
- QRNG (PyPI) integration

```python
from qrand import QiskitBitGenerator
from qiskit import IBMQ

provider = IBMQ.load_account()
bitgen = QiskitBitGenerator(provider)

r = bitgen.random_raw()
s = bitgen.random_bitstring(2)
i = bitgen.random_uint(4)
f = bitgen.random_double(8)

bitgen.load_cache('010010110101')
bitgen.dump_cache(flush=True)
# >>> ...010010110101

print(bitgen.state)
# >>> {'bits':64, 'job_config': ...}
```

QRAND

A python tool (v0.2.0 PyPI)

- Efficiency sampling
- API
- **NumPy integration**
- **Extended functionality**
- QRNG (PyPI) integration

```python
from qrand import QiskitBitGenerator
from qiskit import IBMQ

provider = IBMQ.load_account()
bitgen = QiskitBitGenerator(provider)

from numpy.random import Generator
gen = Generator(bitgen)

binomial = gen.binomial(n, p)
exponential = gen.exponential()
gamma = gen.gamma(shape)
logistic = gen.logistic()
lognormal = gen.lognormal()
pareto = gen.pareto(a)
poisson = gen.poisson()
std_normal = gen.standard_normal()
triangular = gen.triangular(l, m, r)
# ...
```

https://numpy.org/devdocs/reference/random/generator.html#distributions

QRAND

A python tool (v0.2.0 PyPI)

- Efficiency sampling
- API
- NumPy integration
- Extended functionality
- **QRNG (PyPI) integration**

```python
from qrand import QiskitBitGenerator
from qrand import Qrng
from qiskit import IBMQ

provider = IBMQ.load_account()
bitgen = QiskitBitGenerator(provider)
qrng = Qrng(bitgen)


z1 = qrng.get_random_complex_rect()
z2 = qrng.get_random_complex_polar()
x1 = qrng.get_random_double(0, 0.1)
x2 = qrng.get_random_float(-1, 1)
i1 = qrng.get_random_int(0, 4)
i2 = qrng.get_random_int32(0, 10)
```

https://pypi.org/project/qrng/

DEMO

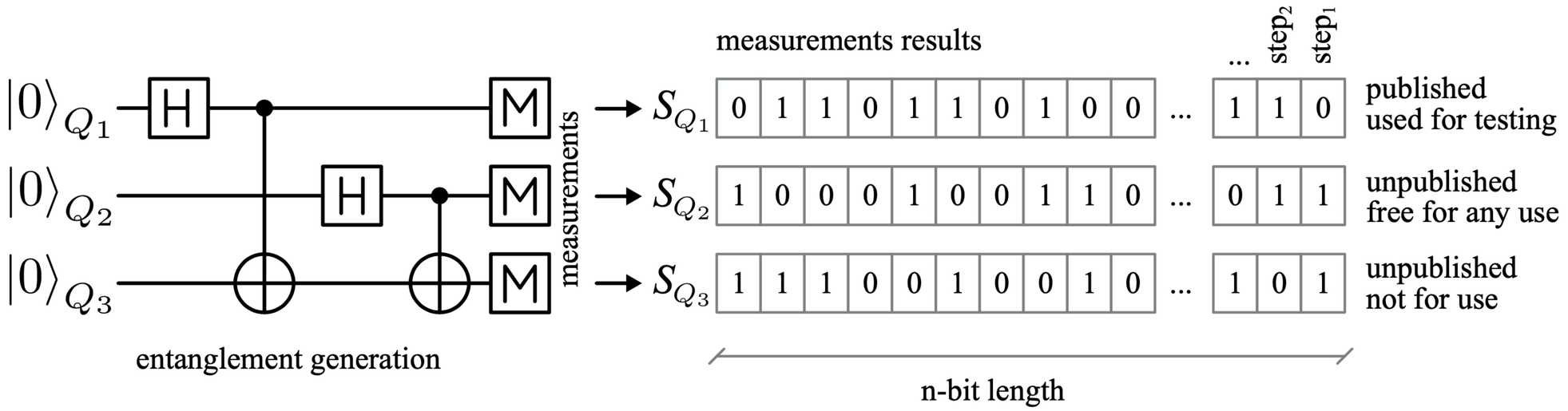
Validation and verification strategies

Validation outsourcing

- Subsequence verification → Exponential overhead

$$|\psi_{Q_1Q_2Q_3}\rangle = \frac{1}{2}\left(|000\rangle + |011\rangle + |101\rangle + |110\rangle\right)$$

TWO BITS OF ENTROPY

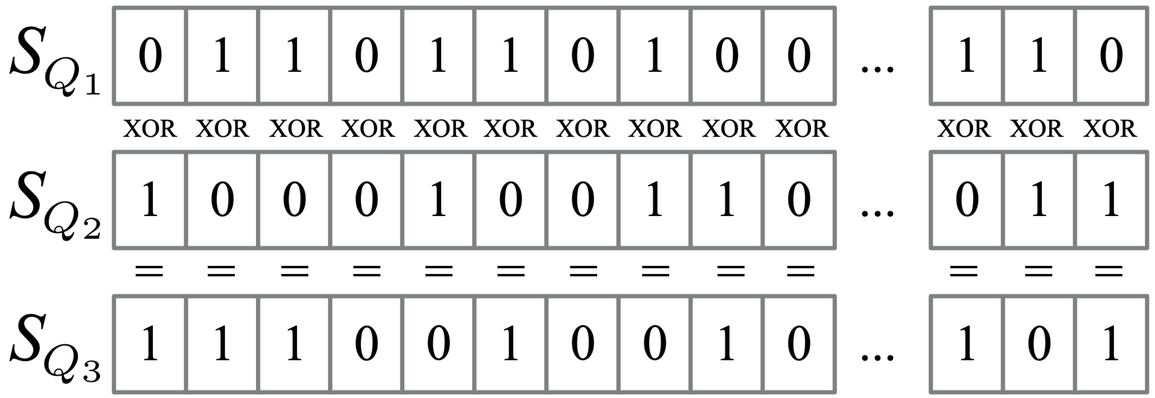


PARITY

Validation and verification strategies

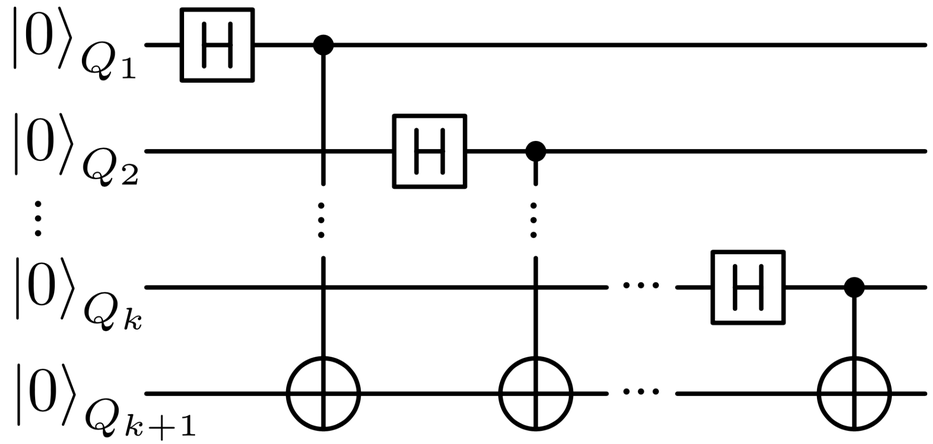
Error correction and scaling

- Parity bit



PURIFY ENTROPY

1 SEQUENCE TO VERIFY

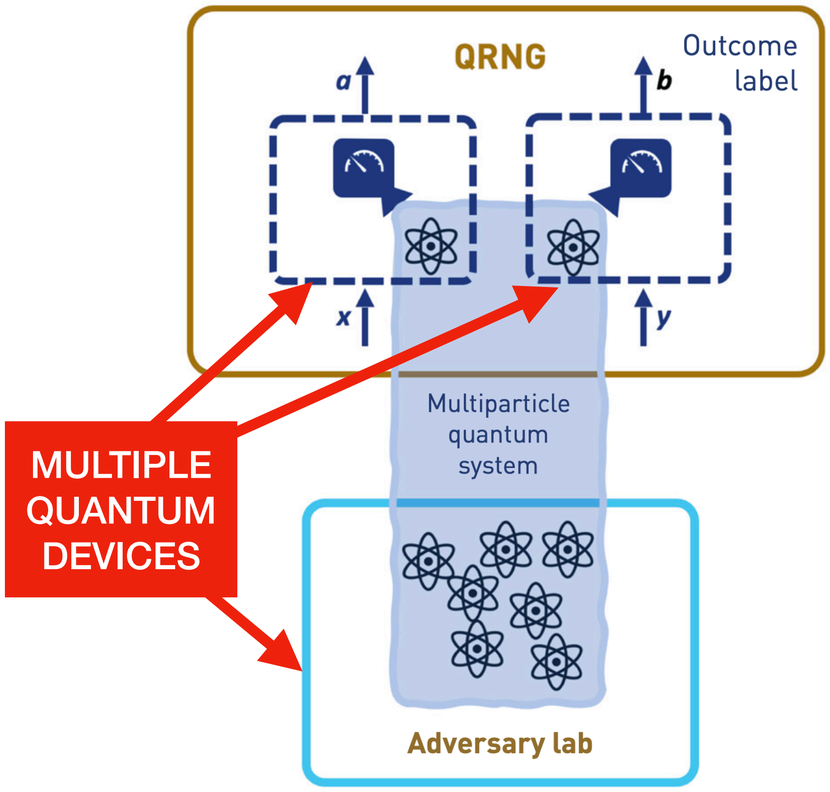


K-1 USABLE SQUENCES

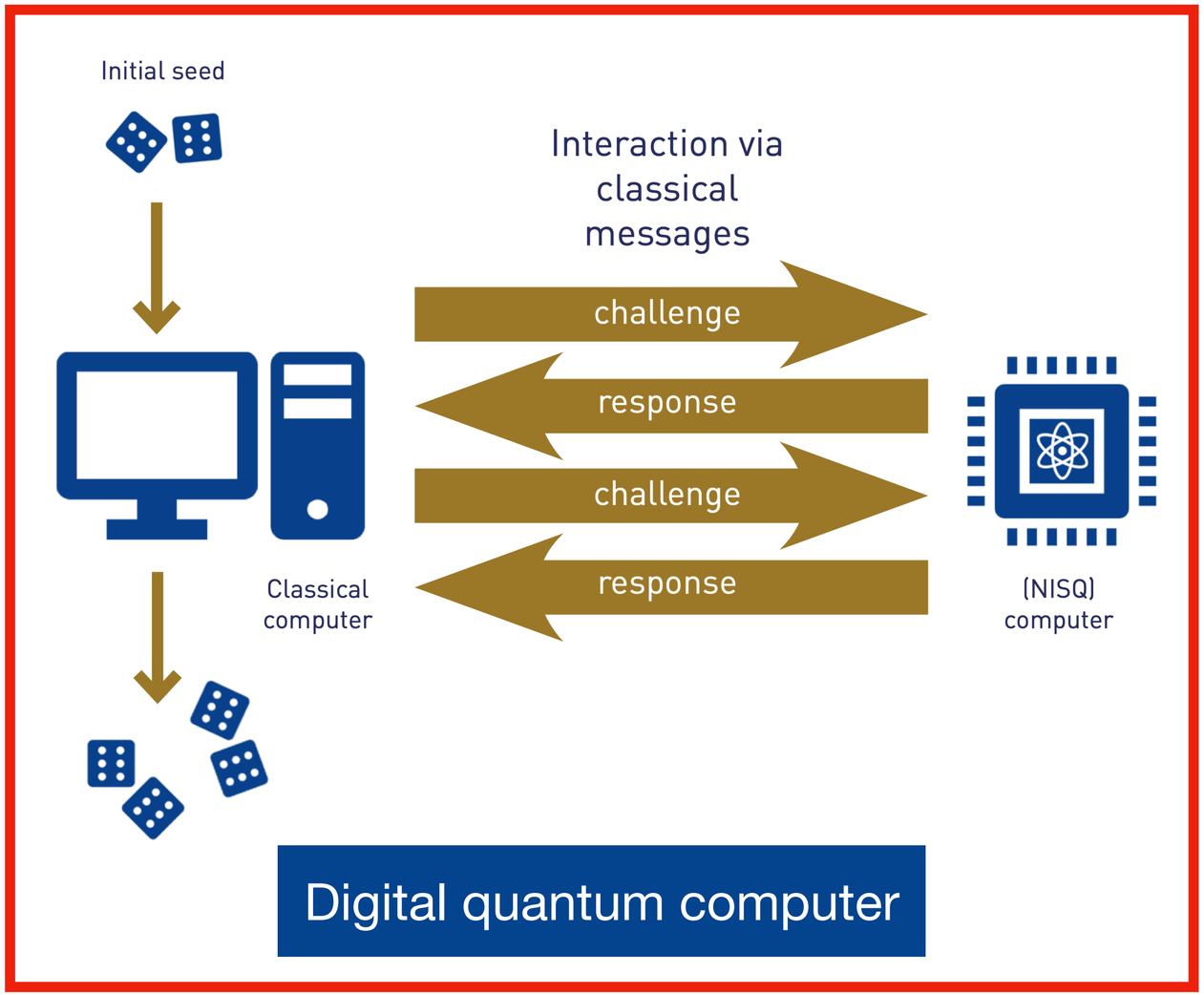
PARITY

Validation and verification strategies

Device independency



Locality proofs

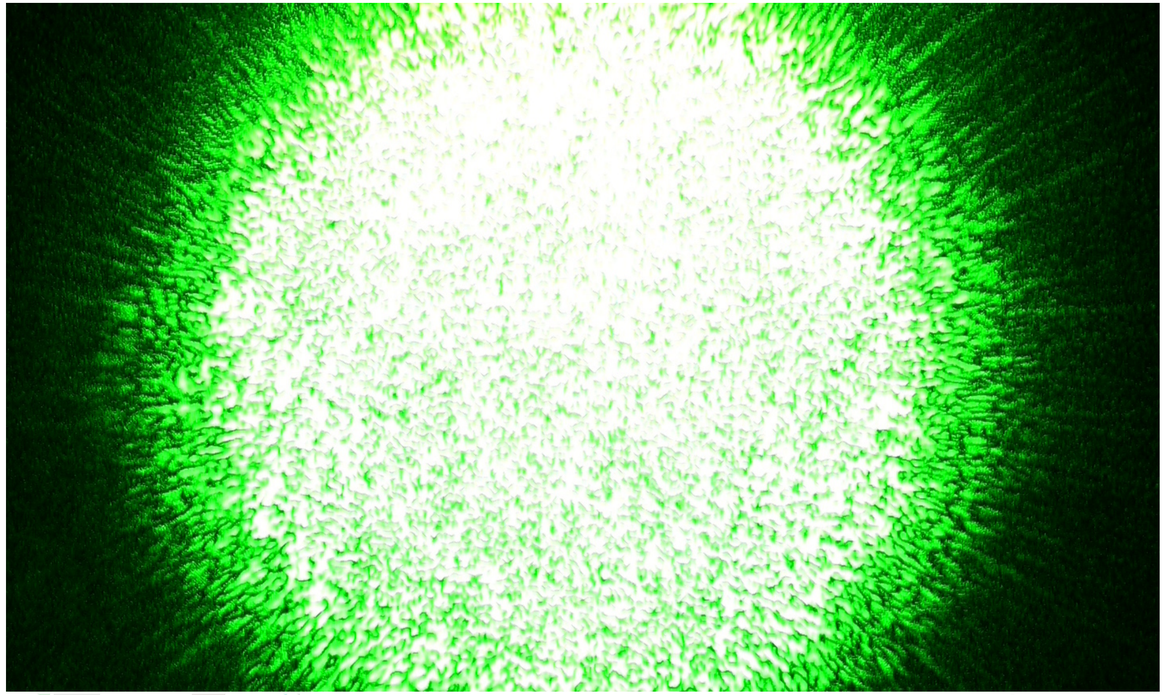


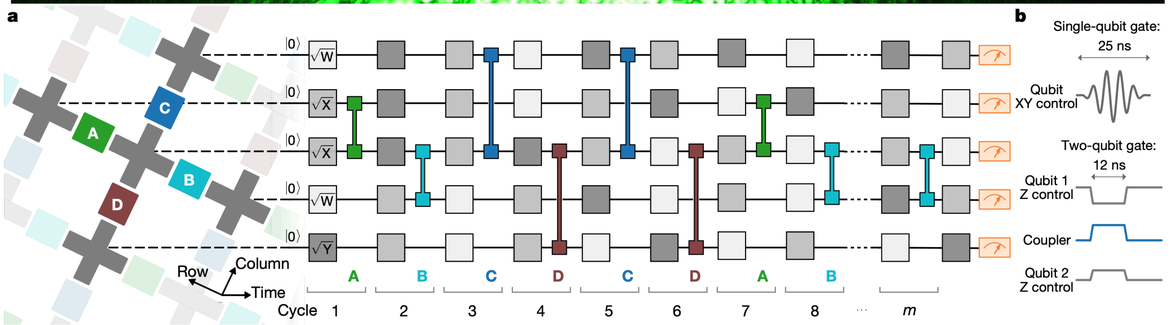
Digital quantum computer

Validation and verification strategies

Google's Sycamore (Scott Aaronson)

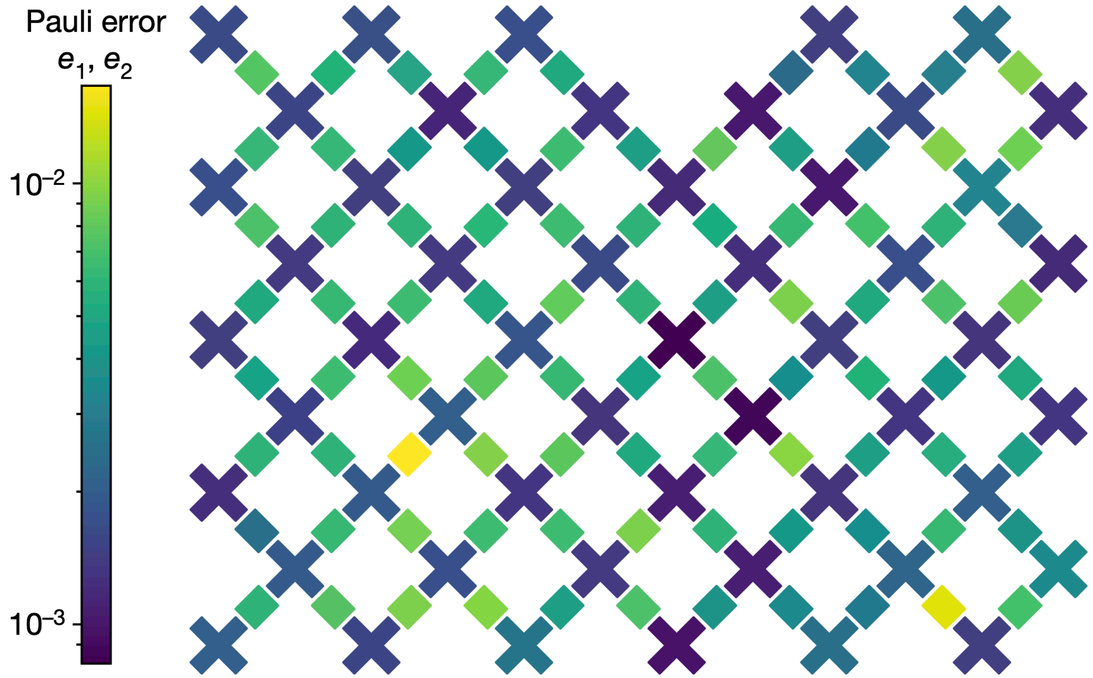
- Process:
  1. *Run a randomly chosen (known) circuit.*
  2. *Make a "small" number of measurements.*
  3. *Simulate results classically and see if they agree.*
- Computational hardness assumption.

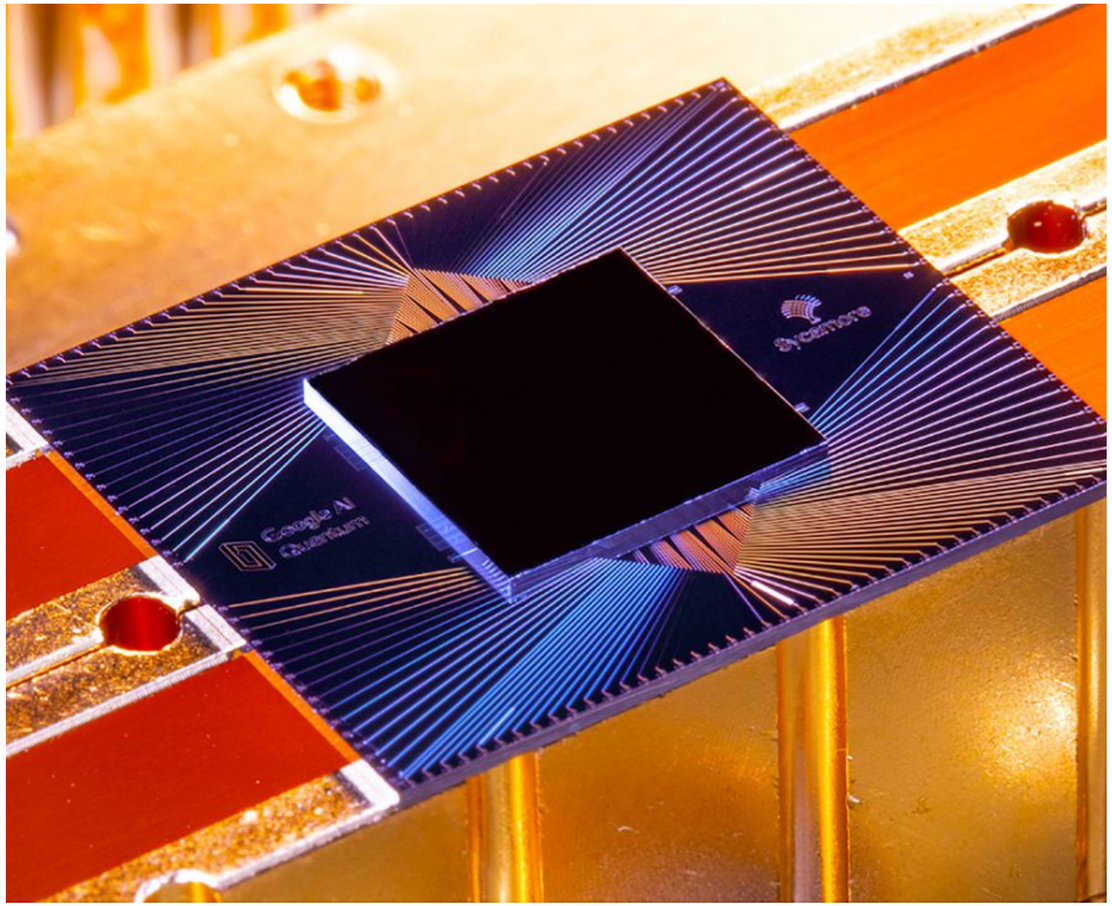




Validation and verification strategies

Quantum supremacy

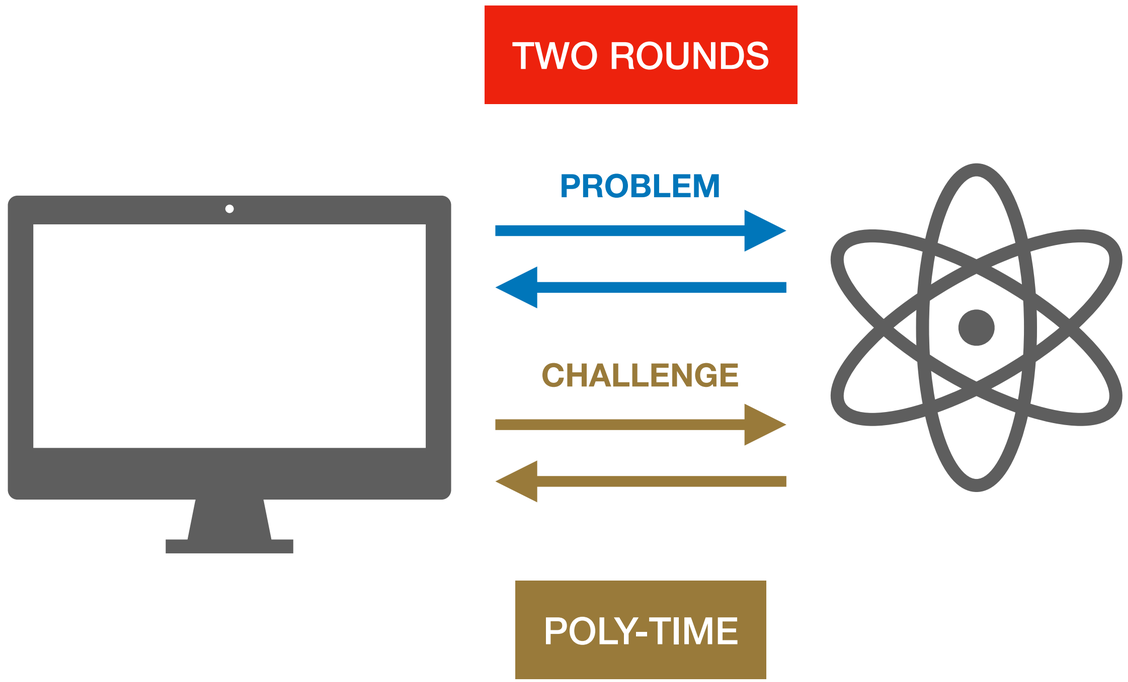




Validation and verification strategies

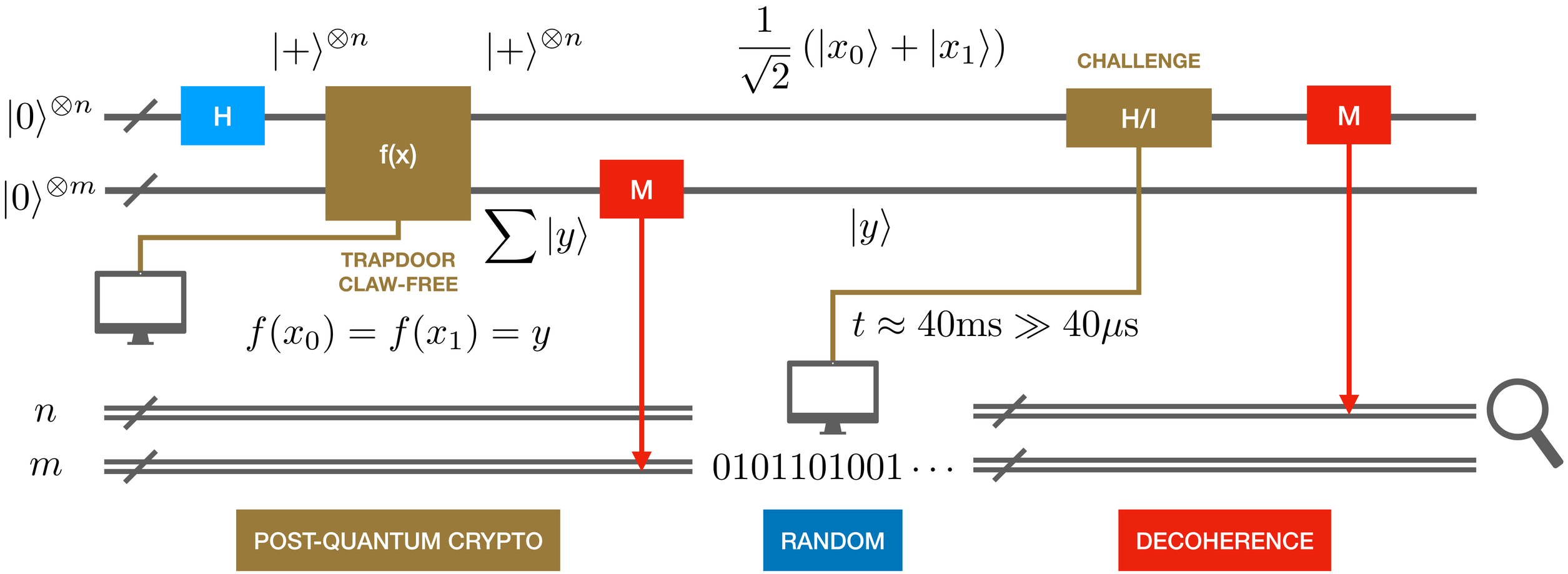
Mahadev's post-quantum cryptography approach

- Previous approach relied on the classical exponential overhead → *NISQ devices*
- **Post-quantum crypto** (not to be confused with quantum crypto)
- Two rounds:
  - *Problem*
  - *Challenge*



Validation and verification strategies

Mahadev's post-quantum cryptography approach



Migration to quantum safe solutions

Why it is important to act now

COMPLIANCE WITH REGULATIONS

NIST standards

LONG MIGRATION TIME

Crypto-agile methodologies

PRODUCTS WITH LONG LIFETIMES

Car industry, self-serving

STORE NOW DECRYPT LATER

Information can be sensitive for extended periods of time

Migration to quantum safe solutions

Building blocks of a crypto-system

- These can all be performed quantumly:

References

(1) Piani et al, *Quantum Random-Number Generators: Practical Considerations and Use Cases.* **EvolutionQ**.

(2) Abellán, *Randomness and quantum entropy w/ Carlos Abellán from Quside.* **Quantum Barcelona**.

(3) Jacak et al, *Quantum random number generators with entanglement for public randomness testing.* **Nature Research**.

(4) Arute et al, *Quantum supremacy using a programmable superconducting processor*. **Nature**.

(5) Brakerski et al, *A Cryptographic Test of Quantumness and Certifiable Randomness from a Single Quantum Device.* **arXiv:1804.00640v3**.

Thanks

Pedro Rivero

ILLINOIS TECH

CHICAGO QUANTUM EXCHANGE

Argonne NATIONAL LABORATORY