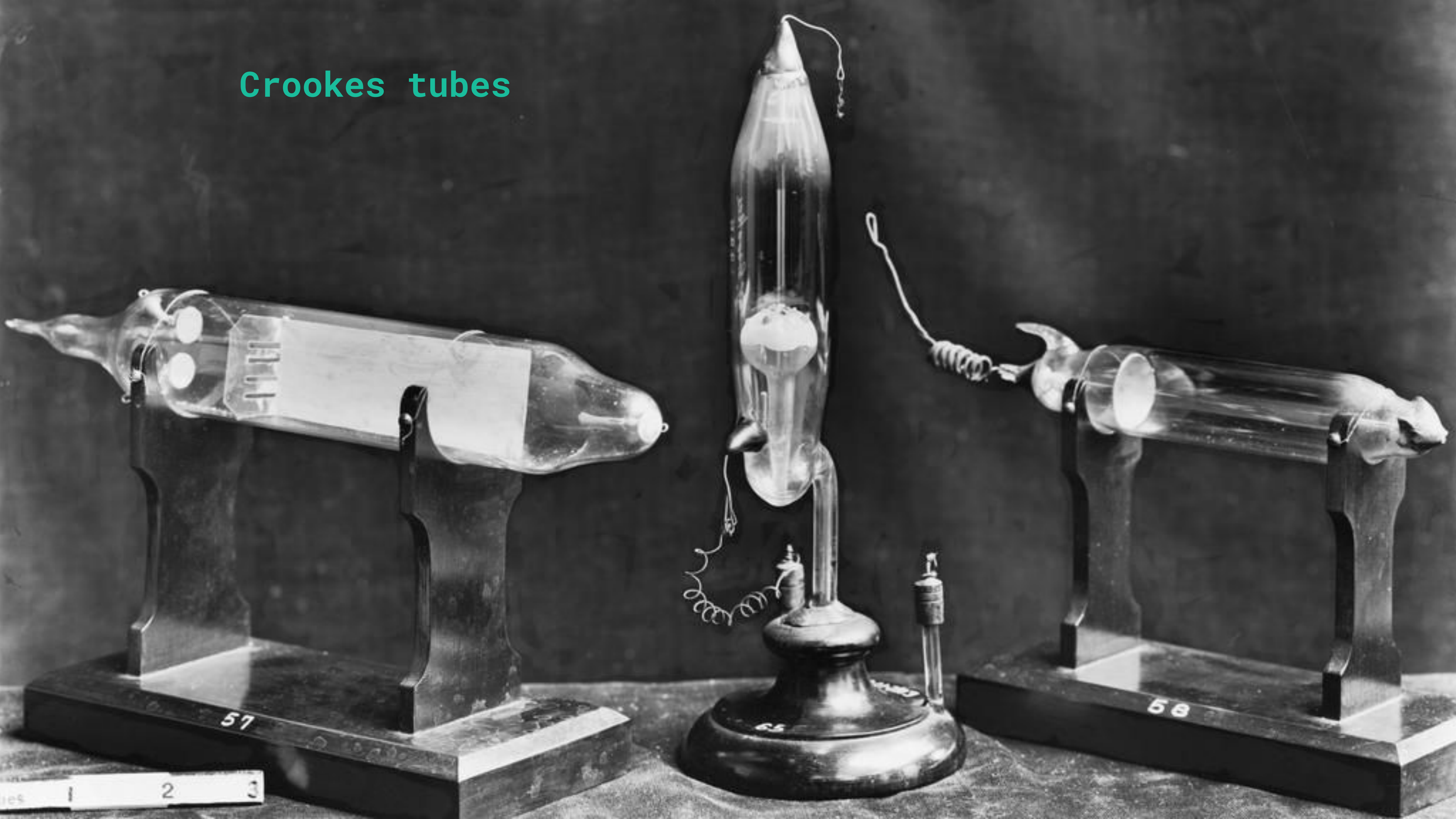
Qubitcoin.
Quantum Proof of Work.

by: superquantum

quantum simulators 🩵 quantum computers

Crookes tubes



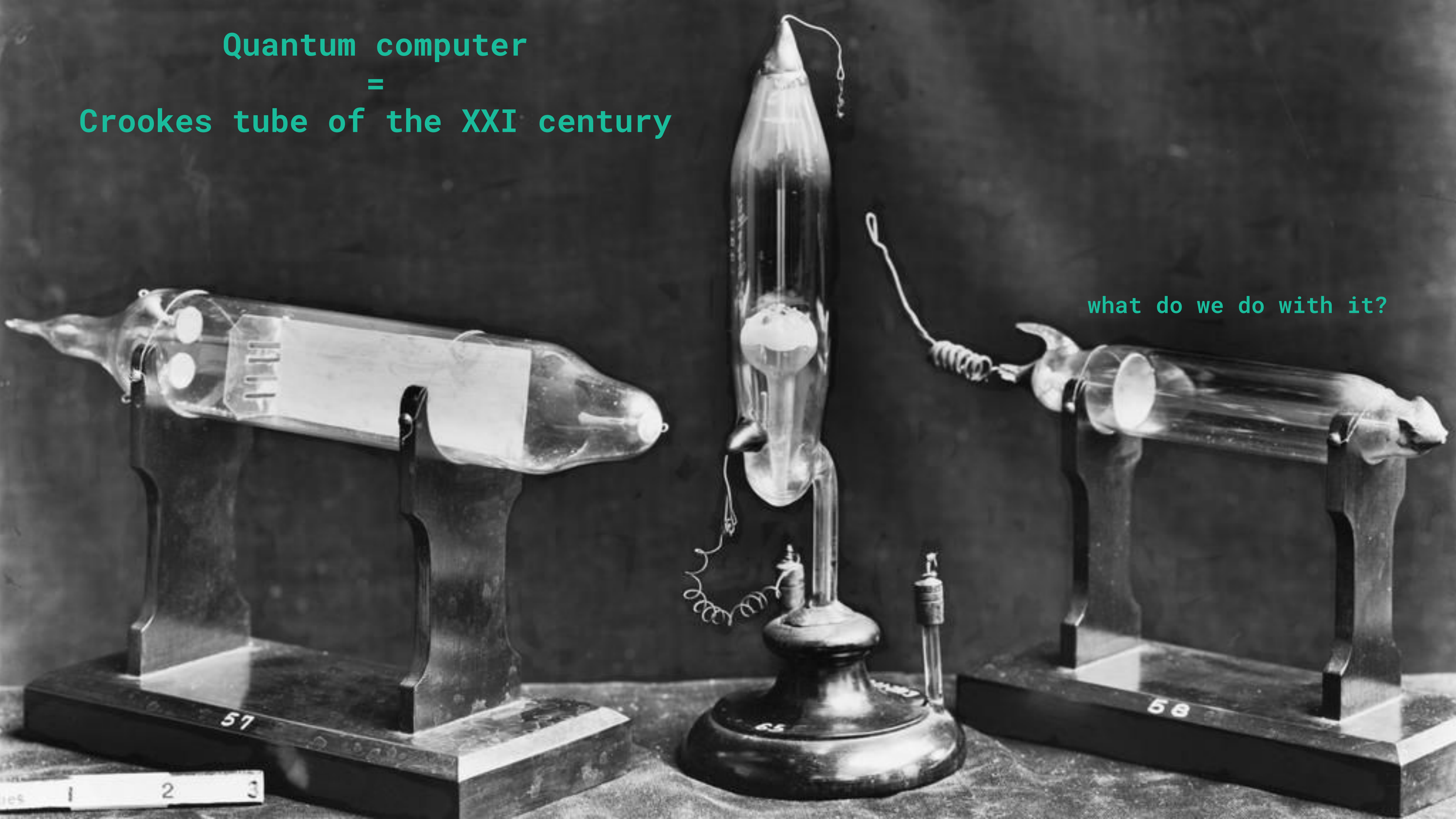
outcomes from crookes tube

1. X-rays
2. Electron discovery and consequent updates in atomic theory
3. Vacuum tubes, early electronics, amplifiers, radio technology
4. Fluorescence, lighting and display devices
5. CRTs,early TV sets and monitors
6. Mass spectrometry

Quantum computer

=

Crookes tube of the XXI century



what do we do with it?

Blockchain

5G

Internet of things

AR/VR

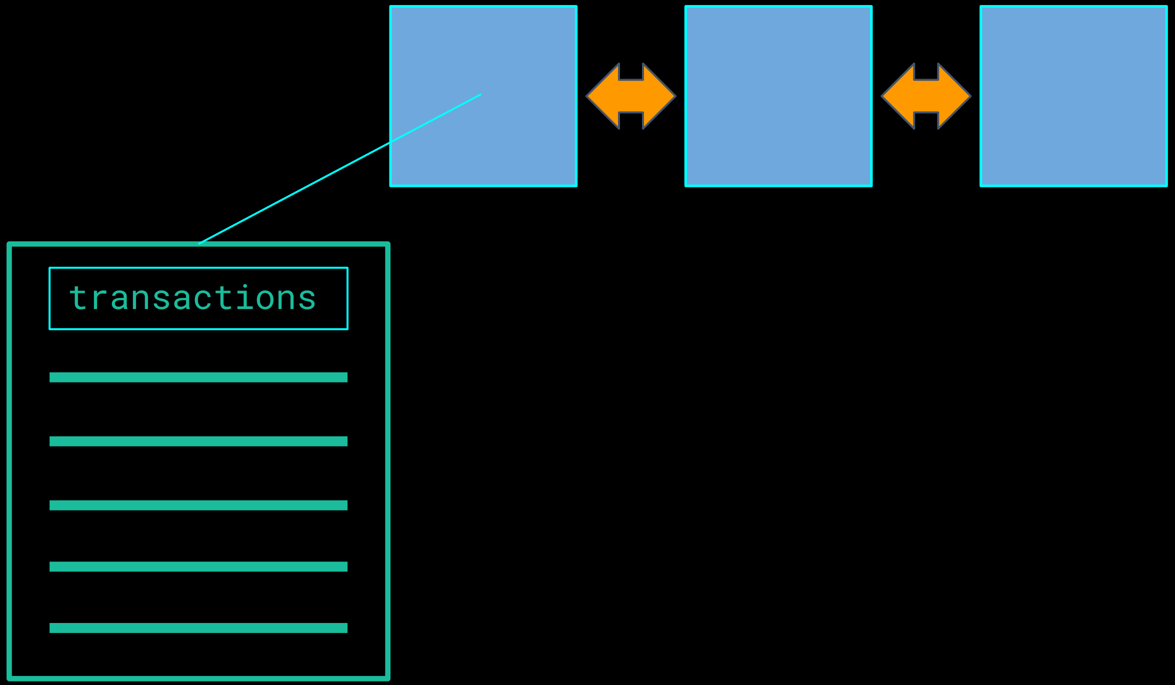
Quantum computing

Nanotechnology

Artificial intelligence

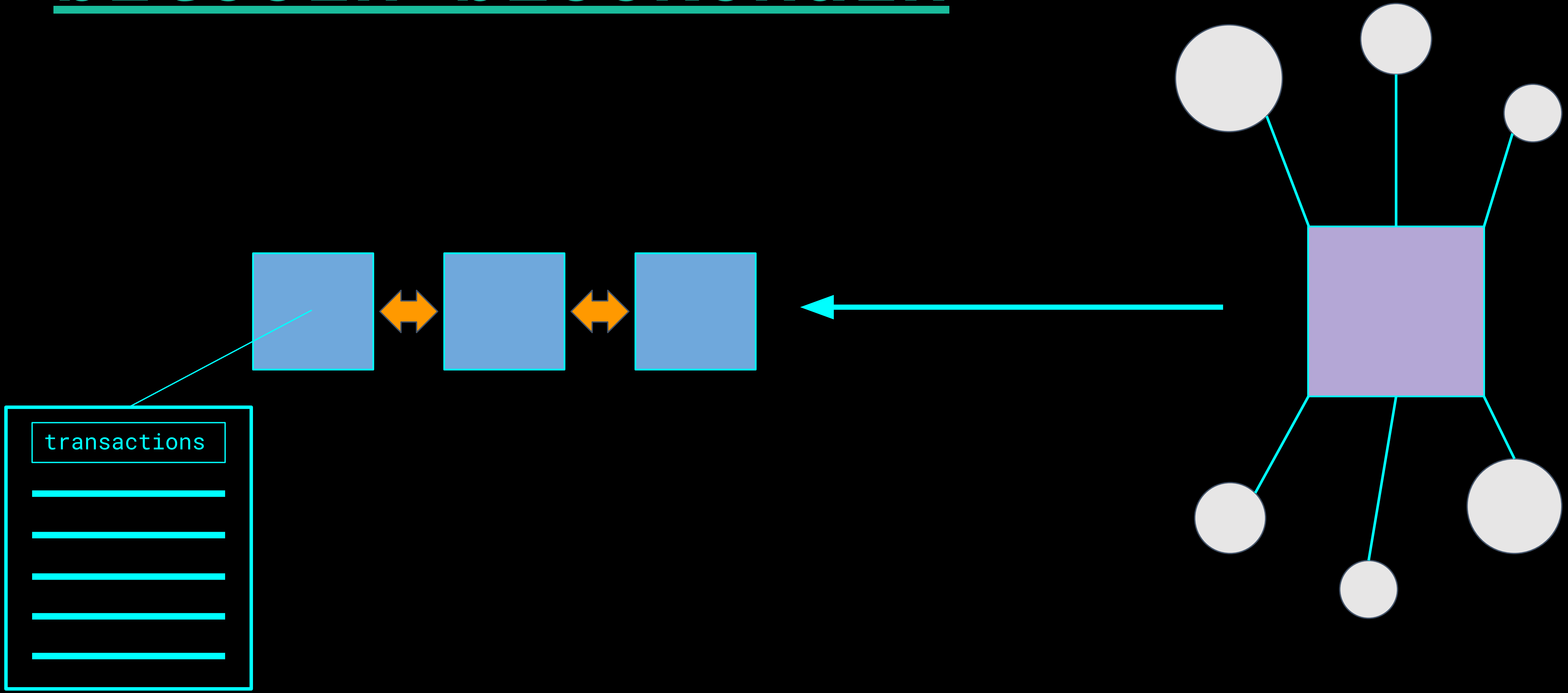
Sustainability & climate change

bitcoin blockchain



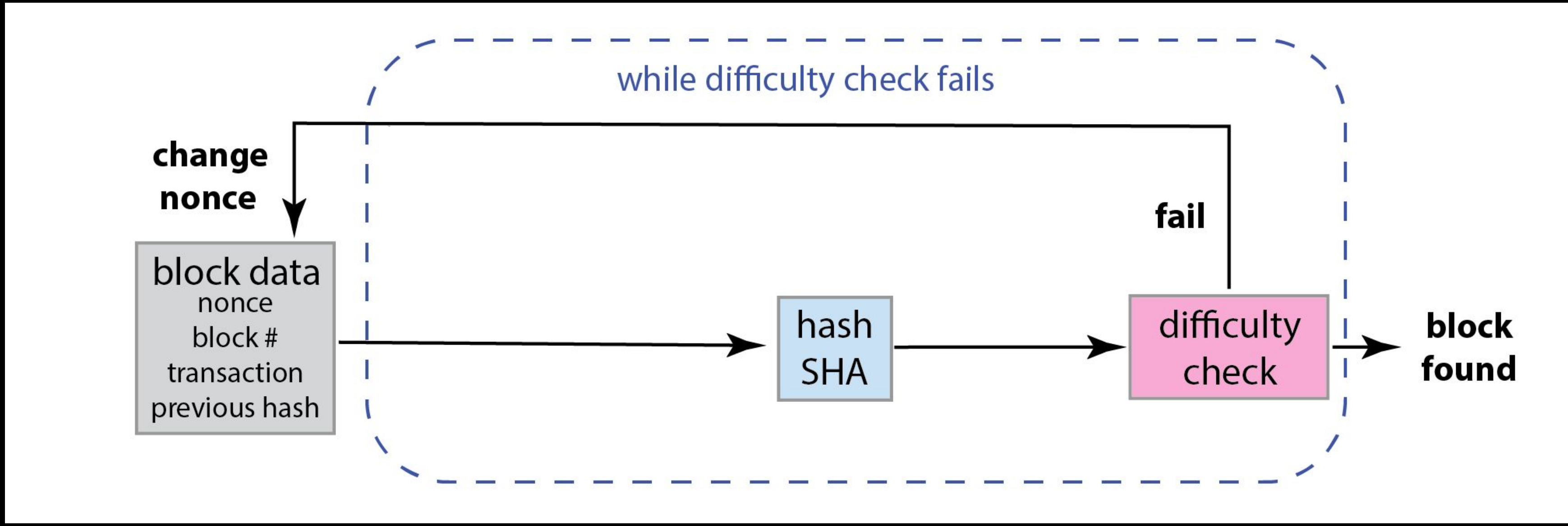
S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System (2008)

bitcoin blockchain



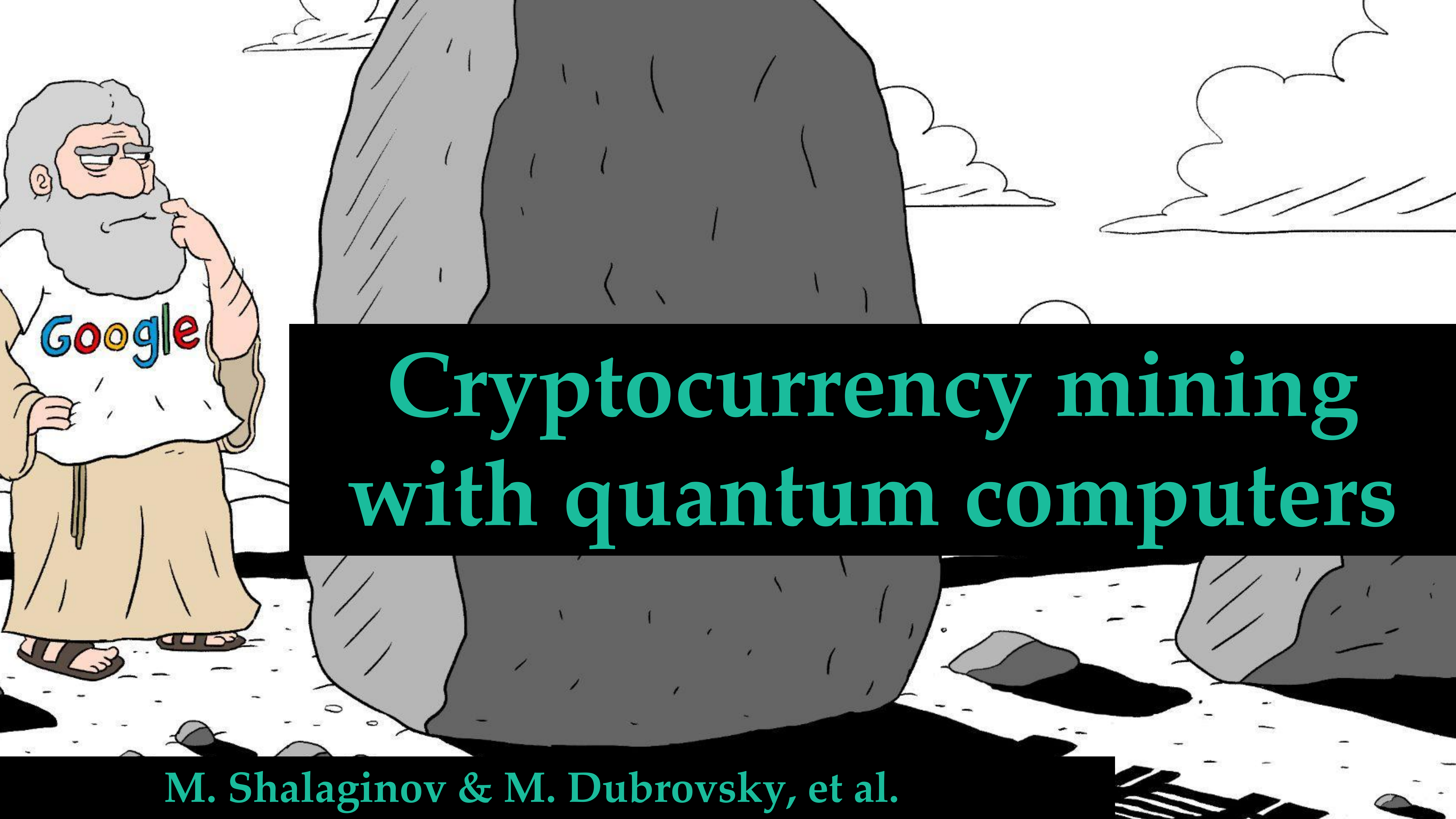
S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System (2008)

Proof of Work (PoW or HashCash)



- HashCash was originally designed to mitigate spam and DoS attacks
- PoW provides a secure and decentralized mechanism for maintaining the integrity of the blockchain ledger

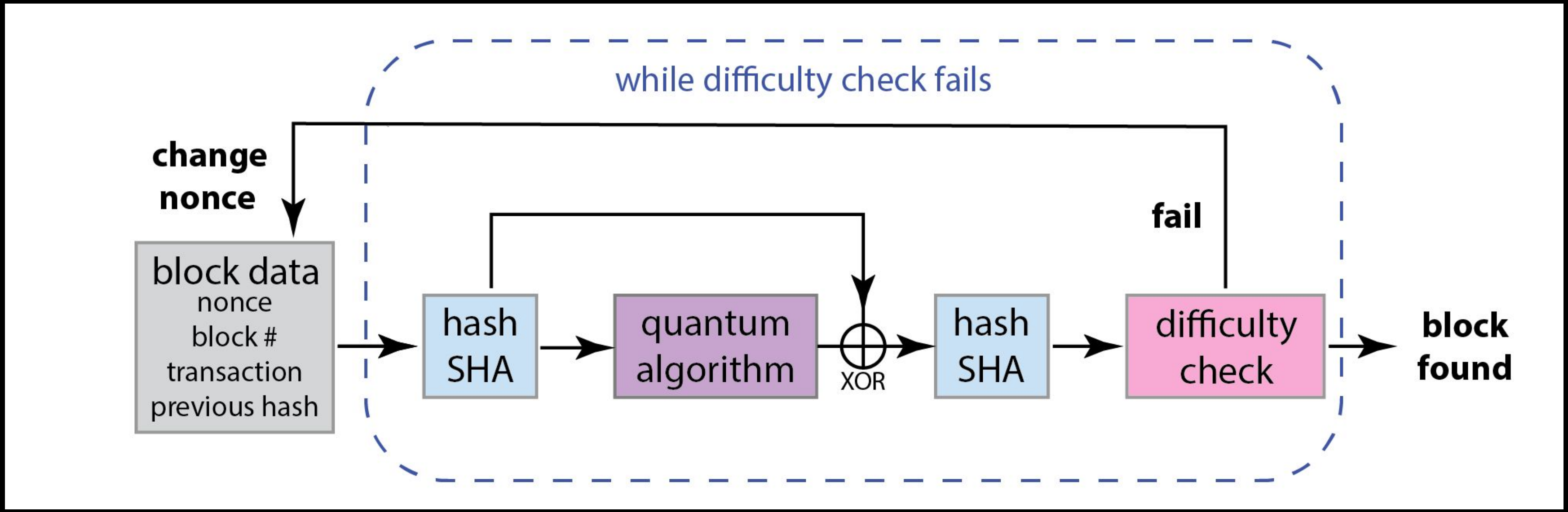
[A. Back, Hashcash-a denial of service counter-measure (2002)]



Cryptocurrency mining with quantum computers

M. Shalaginov & M. Dubrovsky, et al.

quantum Proof of Work

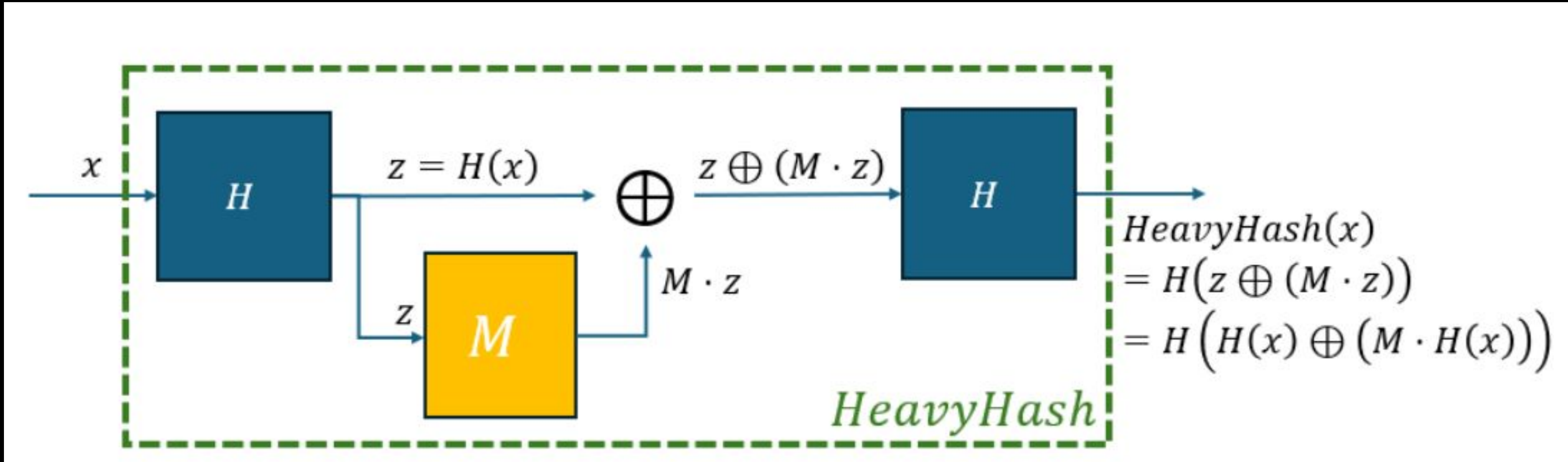


What quantum task to solve?

[M. Shalaginov & M. Dubrovsky, Quantum Proof of Work with Parametrized Quantum Circuits (2022)]

HeavyHash Blockchain

Kaspa with trading volume >= $12B in 24 hours

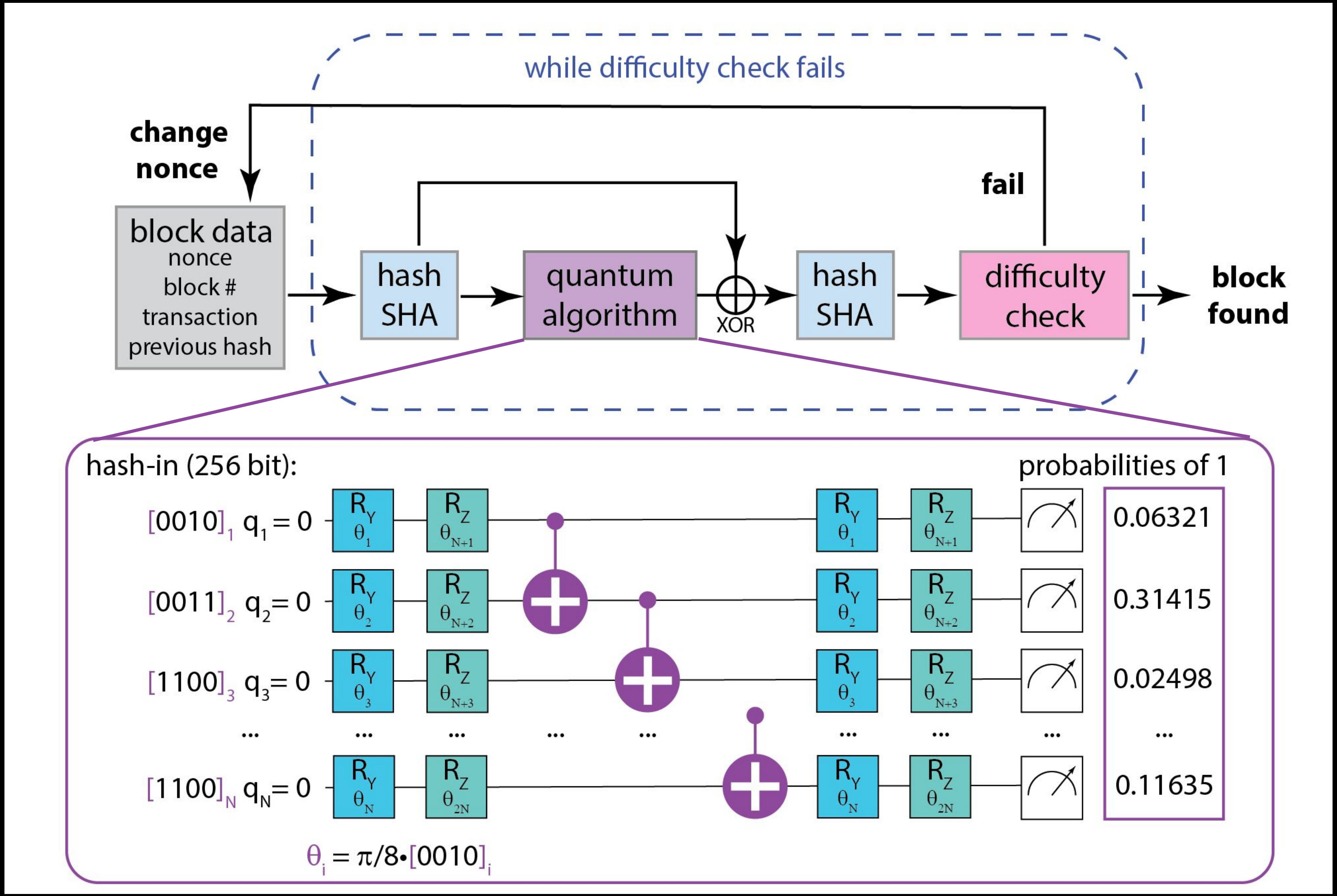


https://kasmedia.com/article/khh-not-broken

HeavyHash originally proposed by PoWx(oBTC): Michael Dubrovsky et al

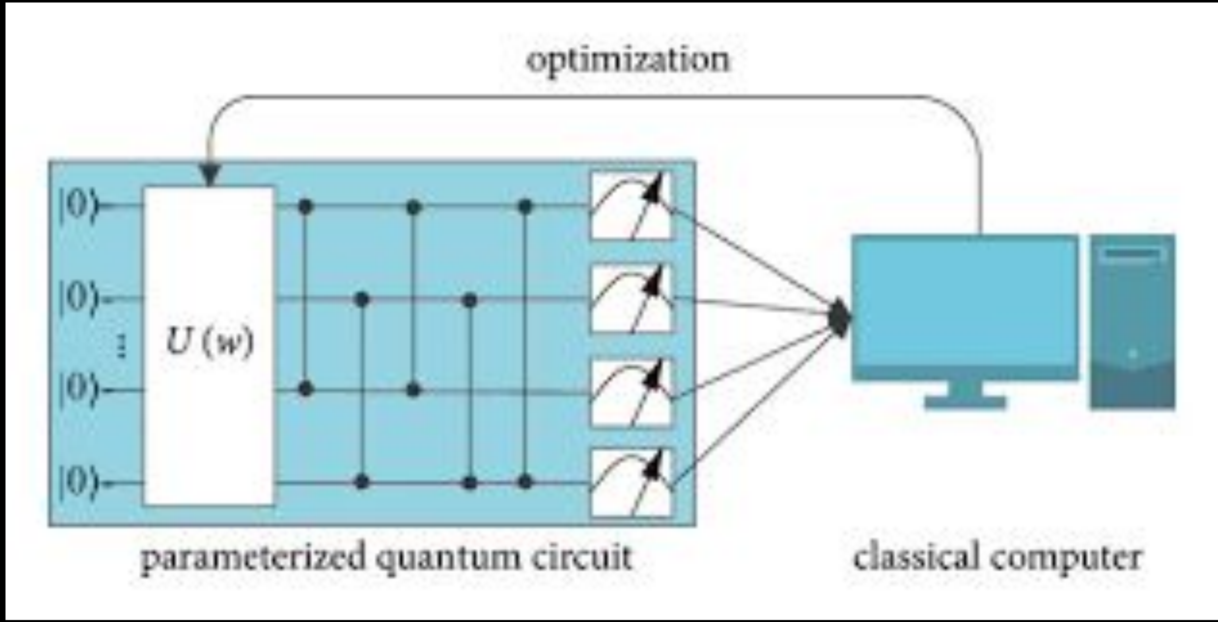
M. Dubrovsky, et al, Towards Optical Proof of Work (2020)

quantum Proof of Work (qHash)



[D. Nizovsky, D. Shatokhin, M. Shalaginov, Boost Quantum Simulators with Your Hashpower (2024)]

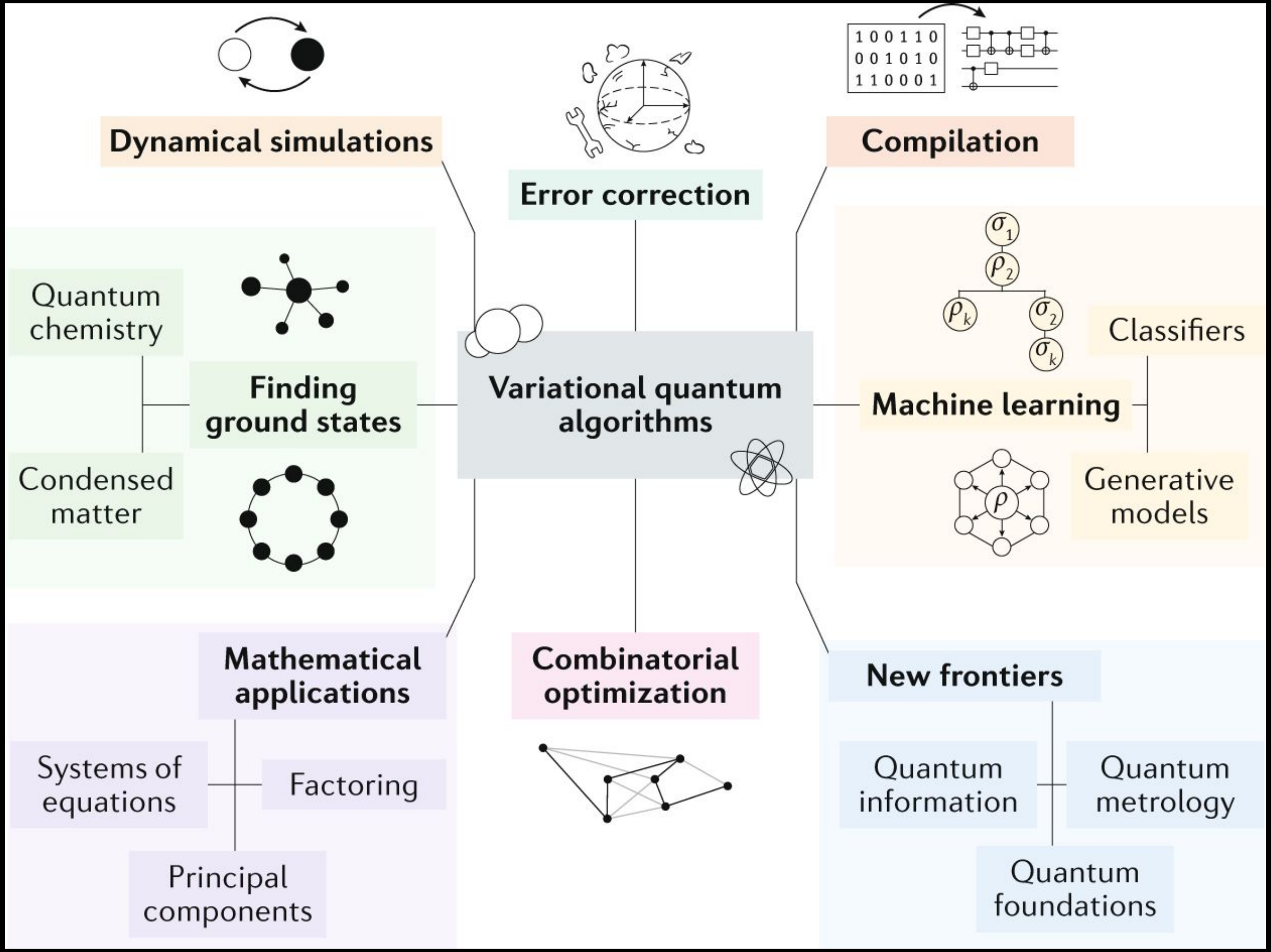
Parameterized Quantum Circuits



Parametrized quantum circuits are the core part of VQAs and enable exploration of the solution space

K. Bharti, et al, Noisy intermediate-scale quantum (NISQ) algorithms (2021)

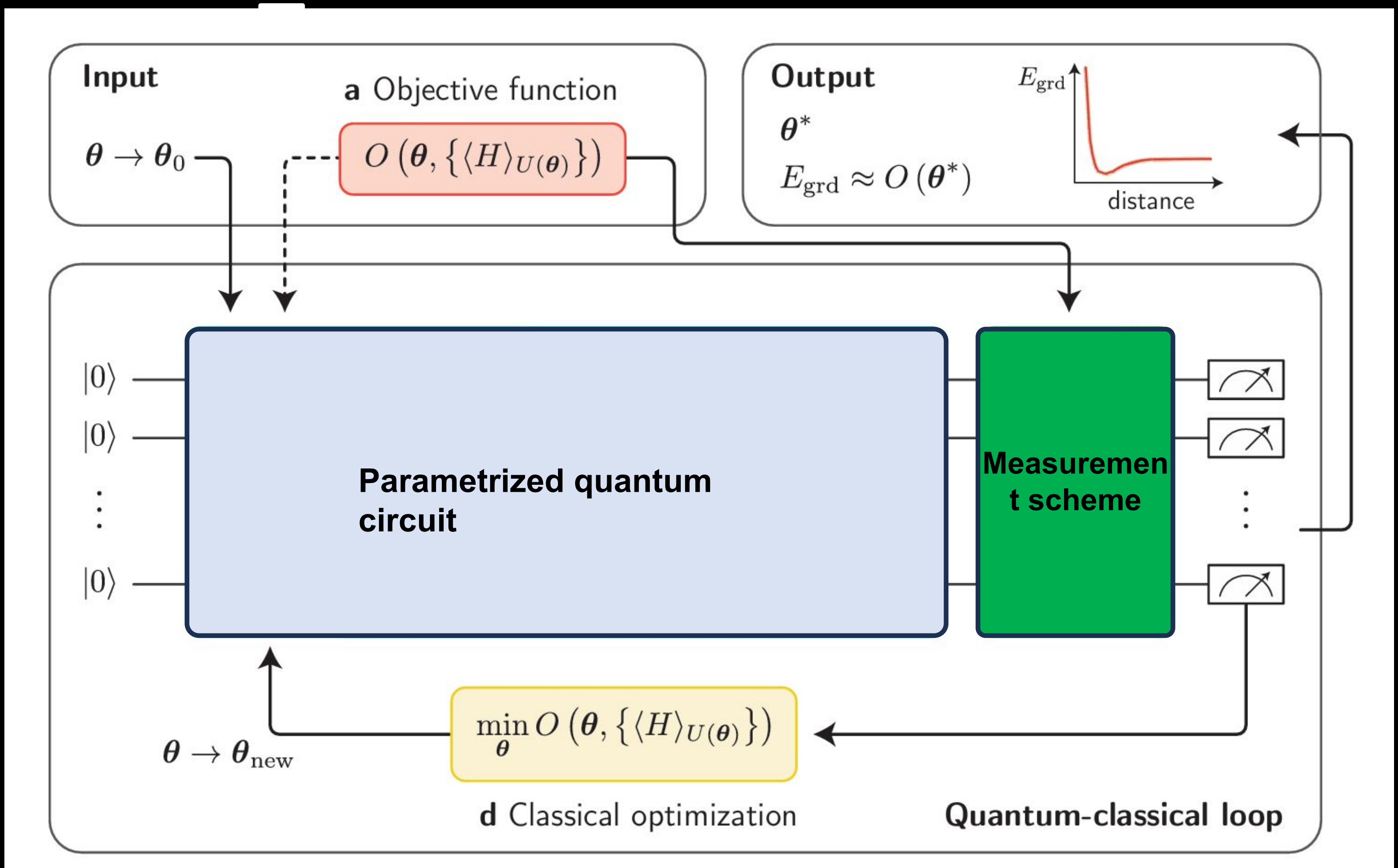
Variational Quantum Algorithms (VQAs)



VQAs are a promising approach to leveraging near-term quantum computers in a wide range of problems

[M. Cerezo, et al, Variational Quantum Algorithms, Nature Reviews Physics (2021)]

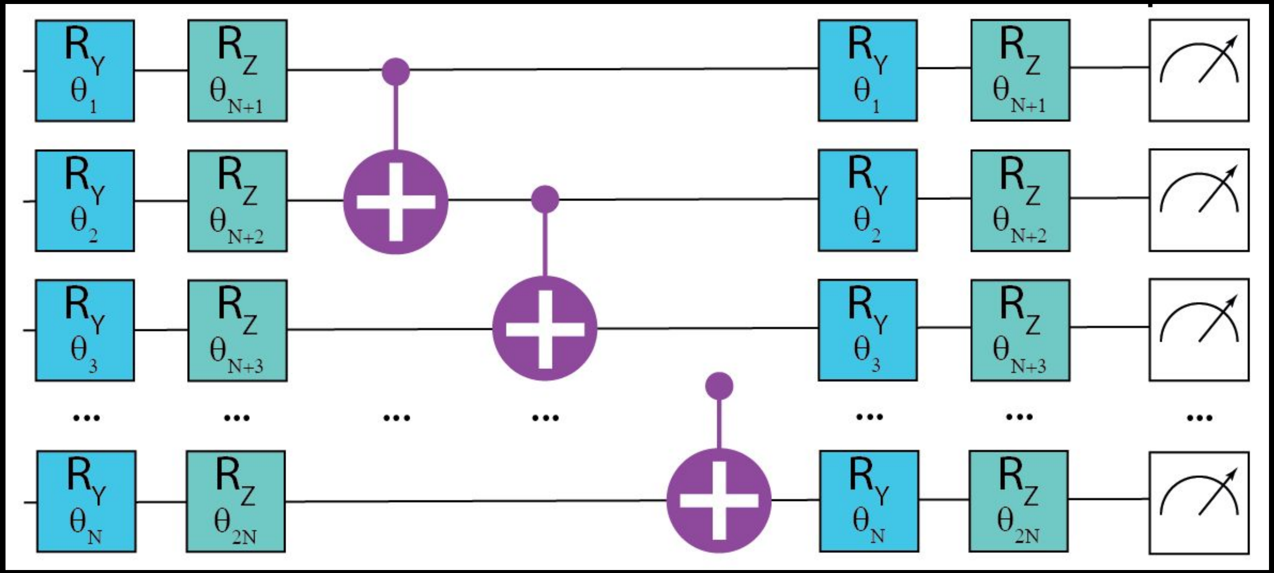
VQA workflow



Parametrized quantum circuits are the core part of VQAs and enable exploration of the solution space

[K. Bharti, et al, Noisy intermediate-scale quantum (NISQ) algorithms (2021)]

Hardware Efficient Ansatz (HEA)



Key features:

- Native set of gates and connectivities
- 1D layers
- Minimal number of parameters
- Noise resilience (quantum hardware)

Parametrized quantum circuits are the core part of VQAs and enable exploration of the solution space

how to start mining?

NOT FOR THE CHALLENGE

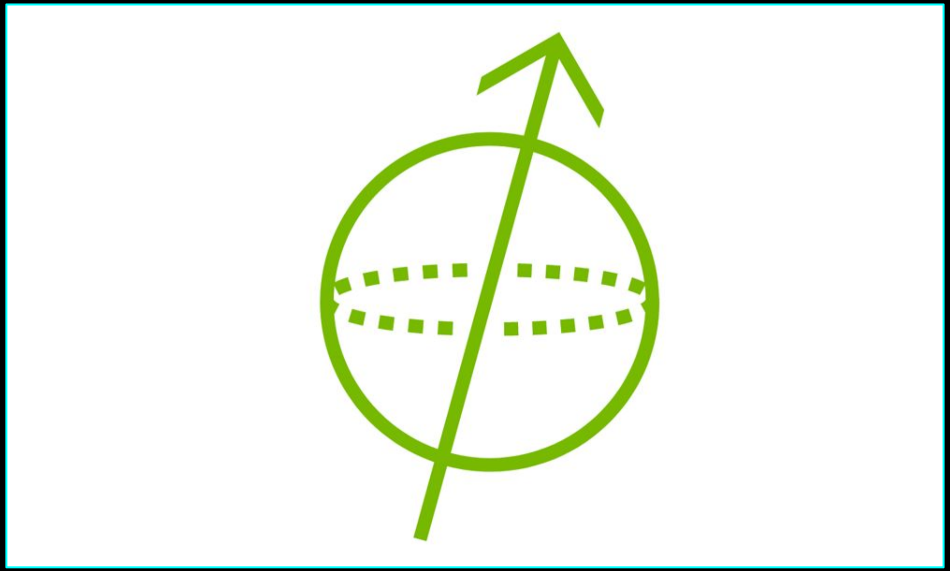
- Linux machine or WSL if on Windows 10,11
- NVIDIA GPU with CUDA compute capability >= 7.0 (can be checked on NVIDIA's website)
- Launch a node and connect it to one of the primary nodes
- Deploy a miner connected to your the node's rpc interface
- See detailed instructions here: https://github.com/super-quantum/qubitcoin/blob/master/README.md

difficulty adjustment

- Number of qubits: 16 (~30 qubits can be computed with a regular GPU)
- Difficulty is adjusted to hashpower, similar as in Bitcoin
- ASERT implementation from Bitcoin Cash

$$\mathrm{target}_{N+1} = \mathrm{target}_{\mathrm{ref}} \exp([t_N - t_{\mathrm{ref}} - (N - h_{\mathrm{ref}})T]/\tau)$$

choosing a simulator



Default - NVIDIA cuStateVec

To integrate with the existing miner:

- Must have C API
- Requires reimplementing 2 functions with the desired simulator

about a challenge: good quantum hash function properties

- Output determinism
- Preservation of entropy
- Computational difficulty
- Preimage resistance
- Collision resistance
- Computational feasibility
- Computation time
- Purely quantum hashing

example hash function

```python
from qiskit import QuantumCircuit
from qiskit.quantum_info import Pauli, Statevector
import numpy as np

def simple_quantum_hash(input_bytes: bytes):
    num_qubits = len(input_bytes)
    qc = QuantumCircuit(num_qubits)
    for i in range(num_qubits):
        angle = (input_bytes[i] / 255) * np.pi  # scale to [0, π]
        qc.rx(angle, i)

    sv = Statevector.from_instruction(qc)
    exp_vals = [sv.expectation_value(Pauli("Z"), [i]).real for i in range(num_qubits)]

    # Map each expectation value from [-1, 1] to an 8-bit integer in [0, 255].
    output_bytes = bytearray([min(int(((val + 1) / 2) * 256), 255) for val in exp_vals])

    return output_bytes
```

hash function analysis

Output determinism

Purely quantum hashing

hash function analysis

```
print(list(simple_quantum_hash(bytes(range(0, 260, 20)))))
```

```
[255, 252, 240, 222, 198, 170, 139, 108, 78, 50, 28, 11, 2]
```

Quantized cosine

hash function analysis

Computational difficulty

Preimage resistance

hash function analysis

```
print(list(simple_quantum_hash(bytes(range(0, 20)))))
print(list(simple_quantum_hash(bytes(range(236, 256)))))
```

```
[255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 254, 254, 254, 254, 253, 253, 253, 252, 252]
[3, 3, 2, 2, 2, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Collision resistance

hash function analysis

Preservation of entropy

```
print(list(simple_quantum_hash(bytes(range(120, 135)))))
```

```
[139, 138, 136, 135, 133, 131, 130, 128, 127, 125, 124, 122, 120, 119, 117]
```

hash function analysis

Computational feasibility - 32 qubits

Computation time

join us!

- repo: github.com/super-quantum
- telegram: @qubitcoingroup
- discord: https://discord.gg/FTmV3GYd9a

- email: qubitcoin@superquantum.io
- website: superquantum.io
- youtube: @mega-super-quantum
- medium: superquantum.medium.com

@QUBITCOINGROUP

we are also looking for blockchain devs and quantum physicists to join our team