Quantum Money and Inflation Control

Final Project Proposal for PHYS C191A

Juncheng Ding, Tian Ariyaratrangsee, Xiaoyang Zheng
University of California, Berkeley – Fall 2025

1. Introduction

Following our first report and critique, this proposal focuses on the simulation and theoretical modeling of **quantum money inflation**. Previous studies identified that while the no-cloning theorem prevents counterfeiting, it does not inherently prevent the *unlimited issuance* of valid quantum states ("quantum inflation"). The project aims to quantitatively demonstrate this inflation effect through simulation and propose a physically realizable method to limit it.

The work builds on Wiesner's *Conjugate Coding* and Zhandry's *Quantum Lightning* frameworks, as well as the complexity analysis described by Aaronson's *The Complexity of Quantum States and Transformations*. By examining how the minting difficulty of quantum tokens scales with quantum computational power, we will simulate the inflationary behavior and test improved models for stability.

2. Project Objectives and Structure

We divide the project into four main parts, with corresponding submilestones:

Part 1: Demonstrating Unlimited Inflation

We will create a simple numerical model simulating the inflationary dynamics of quantum money issuance. The model will assume that the rate of successful minting scales inversely with computational difficulty D, but exponentially with quantum computational speed Q, approximated by

$$R \propto \frac{Q}{D}.$$

As Q grows exponentially, we expect inflation $I \to \infty$ unless compensating mechanisms are applied. This simulation will serve as a quantitative justification for the inflation problem.

subsection*Part 2: Model Design for Limiting Inflation

This section forms the theoretical and computational core of the project. Our objective is to extend existing decentralized quantum money frameworks by introducing a mechanism that quantitatively limits inflation. The design combines theoretical derivation with simulation, connecting system complexity, physical resource constraints, and quantum verification cost.

2.1 Existing Model Analysis

We begin by formalizing the inflation dynamics of Zhandry's *Quantum Lightning* framework. In this model, each unit of currency corresponds to a unique quantum state $|\psi_i\rangle$ whose validity is certified by a publicly verifiable quantum hash operator $\hat{H}$. A state is accepted as genuine when its measurement outcome satisfies $H(|\psi_i\rangle) < D$, where D denotes the difficulty threshold.

However, as the available quantum computing power Q (measured by qubit count or circuit depth) increases, the probability of successfully generating a valid state scales approximately as $P \sim \frac{Q}{D}$. If D remains constant, the minting rate $R = \alpha P$ grows exponentially, leading to an unbounded monetary supply $M(t) \sim e^{\beta t}$ and resulting in runaway "quantum inflation." Our initial simulation will reproduce this scaling relationship and demonstrate the divergence that occurs when difficulty adjustment lags behind computational advances.

2.2 Improved Model Proposal: Resource Token Mechanism

This section constitutes the central technical innovation of our project. To mitigate the exponential inflation growth, we propose a **Resource Token (RT) mechanism** that couples quantum state generation to bounded physical resources, creating an economic scarcity layer independent of computational advances.

Core Principle: Hilbert Space Dimensionality as Monetary Cost The fundamental insight is to link minting cost directly to the intrinsic quantum complexity of the monetary state. For Zhandry's Quantum Lightning construction with parameters (n, k, m), each valid bolt $|\psi_y\rangle$ lives in a Hilbert space of dimension $d = 2^m$. We propose that generating such a state must consume a **Resource Token (RT)** proportional to its information-theoretic content:

$$\mathrm{RT}_{\mathrm{cost}}(|\psi_y\rangle) = \kappa \cdot \log_2(d) = \kappa \cdot m,$$

where $\kappa > 0$ is a system-wide scaling constant governing monetary policy.

The total RT supply is bounded: $\mathrm{RT}_{\mathrm{total}} = R_{\max}$, establishing a hard cap on concurrent monetary states. The modified minting protocol becomes:

$$\text{Accept } |\psi\rangle \iff \begin{cases} H(|\psi\rangle) < D & \text{(validity check)} \\ \mathrm{RT}_{\mathrm{available}} \geq \mathrm{RT}_{\mathrm{cost}}(|\psi\rangle) & \text{(resource check)} \end{cases}$$

Inflation Bound Analysis Under this mechanism, the maximum achievable money supply satisfies:

$$M_{\max} = \frac{R_{\max}}{\langle \mathrm{RT}_{\mathrm{cost}} \rangle} = \frac{R_{\max}}{\kappa \cdot m},$$

where $\langle \mathrm{RT}_{\mathrm{cost}} \rangle$ is the average RT cost per bolt. Critically, even if quantum computational power $Q(t)$ grows exponentially, the minting rate is bounded:

$$R_{\mathrm{bounded}}(t) \leq \frac{R_{\max}}{T_{\mathrm{mint}}(Q)} \cdot \frac{1}{\mathrm{RT}_{\mathrm{cost}}} = O(1),$$

where $T_{\mathrm{mint}}(Q)$ is the wall-clock time per successful mint. This achieves *asymptotic inflation control*: $\lim_{t\to\infty} M(t) = M_{\max} < \infty$.

Three Technical Implementation Pathways We investigate three concrete mechanisms for enforcing RT consumption in quantum circuits:

Pathway A: Gate-Count-Dependent Penalty

Associate each quantum gate with a fixed RT cost ϵ_{gate}. For a minting circuit with total gate count G and depth L, define:

$$\mathrm{RT}_{\mathrm{cost}} = \alpha \cdot G + \beta \cdot L,$$

where α, β are tunable parameters balancing spatial vs. temporal resource usage. This approach is *circuit-agnostic* and easily implementable in existing quantum frameworks (Qiskit, Cirq).

Advantages: Simple to track; compatible with existing compilers. *Limitations:* Does not capture semantic complexity; vulnerable to circuit optimization exploits.

Pathway B: Decoherence-Aware Token Consumption

Exploit quantum decoherence as a natural "cost" mechanism. Define RT consumption as a function of cumulative decoherence probability:

$$\mathrm{RT}_{\mathrm{cost}} = \gamma \cdot \int_0^{T_{\mathrm{exec}}} \Gamma(t)\, dt,$$

where $\Gamma(t)$ is the instantaneous decoherence rate and T_{exec} is circuit execution time. Longer, deeper circuits experience higher decoherence $\rightarrow$ consume more RT $\rightarrow$ naturally limit complex state generation.

Advantages: Physically grounded; automatically adapts to hardware quality. *Limitations:* Requires accurate noise modeling; may disadvantage low-quality hardware.

Pathway C: Ancilla Qubit Pool Depletion (Most Physically Realistic)

Introduce a finite, non-replenishable pool of N_{ancilla} ancilla qubits. Each minting operation permanently consumes a ancilla qubits for error correction or state verification:

$$\text{RT}_{\text{cost}} = a \cdot c_{\text{ancilla}},$$

where c_{ancilla} is the "worth" of each ancilla. When the ancilla pool is exhausted ($N_{\text{remaining}} = 0$), no further minting is physically possible.

Advantages: Corresponds to real hardware constraints; unambiguous resource accounting. *Limitations:* Requires modified circuit architecture; ancilla reuse protocols complicate accounting.

Comparative Evaluation Criteria We will evaluate each pathway against:

- **Enforceability:** Can the mechanism be reliably enforced without centralized oversight?
- **Fairness:** Does it disadvantage certain hardware architectures or quantum algorithms?
- **Robustness:** Is it resistant to gaming or circumvention strategies?
- **Implementability:** Can it be deployed on near-term quantum devices (NISQ era)?

Algorithmic Protocol The complete minting-with-RT protocol proceeds as follows:

1. **Pre-Execution Check:** Query global RT ledger: $\text{RT}_{\text{avail}} \leftarrow \text{Read}(\text{Ledger})$. Estimate circuit cost: $\text{RT}_{\text{est}} \leftarrow \text{EstimateCost}(\text{Circuit})$. If $\text{RT}_{\text{avail}} < \text{RT}_{\text{est}}$, abort with `INSUFFICIENT_RESOURCES`.
2. **Quantum Execution:** Run bolt generation circuit U_{mint} on quantum hardware. Measure output to obtain candidate bolt $|\psi_{\text{cand}}\rangle$ and serial number s.
3. **Verification:** Apply Zhandry's verification protocol $V(|\psi_{\text{cand}}\rangle, s)$. If rejected, refund partial RT cost (e.g., $0.5 \cdot \text{RT}_{\text{est}}$) and terminate.
4. **RT Consumption:** Compute actual cost: $\text{RT}_{\text{actual}} \leftarrow \text{MeasureCost}(\text{Execution})$. Update ledger atomically: $\text{RT}_{\text{avail}} \leftarrow \text{RT}_{\text{avail}} - \text{RT}_{\text{actual}}$. Broadcast new bolt $(s, |\psi_{\text{cand}}\rangle)$ to network.
5. **Difficulty Adjustment:** Every ΔT time units, adjust difficulty $D \leftarrow f(R_{\text{observed}}, R_{\text{target}})$ to stabilize issuance rate.

Security Preservation A critical concern: does the RT mechanism introduce new attack vectors or weaken Zhandry's security proof?

Theorem (Informal): If the underlying Quantum Lightning scheme satisfies $(2k+2)$-NAMCR (non-affine multi-collision resistance), then the RT-augmented scheme preserves uniqueness, provided:

1. RT consumption is monotonic: more complex states cost more RT.
2. The RT ledger update is atomic and append-only (blockchain-enforced).
3. Adversaries cannot forge RT without quantum work.

Proof Sketch: An adversary producing two bolts with identical serial numbers must either: (a) Clone an existing bolt (violates no-cloning), (b) Generate $2(k+1)$ non-affine collisions (violates NAMCR), or (c) Forge RT ledger entries (violates blockchain integrity). None of these are computationally feasible under standard assumptions.

We will formalize this argument and simulate potential attacks (RT exhaustion DoS, ledger manipulation) in Part 3.

2.3 Physical Feasibility: Bridging Theory and NISQ Hardware

This section addresses the practical implementability of our RT mechanism on near-term quantum devices.

Challenge: Mapping Abstract RT to Physical Constraints Current quantum computers (IBM Quantum, Google Sycamore, IonQ) operate with:

- Limited qubit counts: $n_q \in [50, 1000]$ (vs. theoretical requirement $m = 2n^2 \approx 200$ for $n = 10$)
- High error rates: $\epsilon_{\text{gate}} \sim 10^{-3}$ to 10^{-2} (vs. fault-tolerant threshold $\sim 10^{-4}$)
- Finite coherence times: $T_2 \sim 100\mu s$ (limits circuit depth)
- Non-uniform connectivity: constrains circuit topology

Our strategy: *demonstrate proof-of-concept with miniaturized parameters*, then extrapolate scaling behavior.

Miniaturized Quantum Lightning: Toy Implementation For Qiskit simulation, we propose:

$$\begin{cases} n = 3 & \text{(security parameter - toy value)} \\ k = 2 & \text{(collision set size)} \\ m = 12 & \text{(input dimension, manageable for simulation)} \end{cases}$$

This yields:

- Bolt Hilbert space dimension: $d = 2^{12} = 4096$
- Number of bolt copies: $k + 1 = 3$
- Total qubits needed: $3 \times 12 = 36$ (within NISQ reach)
- Estimated gate count: $G \sim O(n^2 m) \sim 100$ gates per mini-verification

RT Implementation: Three Concrete Protocols Protocol 1: Gate-Count Accumulator (Simplest)

Instrument Qiskit's transpiler to track gate operations:

```
class RTTracker:
    def __init__(self, initial_budget):
        self.budget = initial_budget
        self.cost_map = {'cx': 10, 'u3': 1, 'measure': 5}

    def charge_gate(self, gate_name):
        cost = self.cost_map.get(gate_name, 1)
        if self.budget < cost:
            raise InsufficientRTError
        self.budget -= cost
```

Execute minting circuit with RT tracking:

```
tracker = RTTracker(initial_budget=1000)
qc = generate_bolt_circuit(n=3, k=2, m=12)
transpiled = transpile(qc, basis_gates=['u3','cx'])
for gate in transpiled.data:
    tracker.charge_gate(gate[0].name)
result = execute(transpiled, backend).result()
```

Validation: Compare RT consumption across different bolt generation attempts. Expected variance $\sigma_{\text{RT}} < 10\%$ (low variance confirms deterministic cost).

Protocol 2: Decoherence-Weighted RT

Model decoherence using Qiskit Aer's noise models:

```
from qiskit.providers.aer.noise import NoiseModel
noise_model = NoiseModel.from_backend(ibmq_device)

# RT cost = integrated decoherence probability
def compute_RT_decoherence(circuit, noise_model):
    noisy_result = execute(circuit, Aer.get_backend('qasm'),
                           noise_model=noise_model).result()
    fidelity = state_fidelity(ideal_state, noisy_result)
    RT_cost = -log(fidelity) * gamma  # gamma: scaling factor
    return RT_cost
```

Key Insight: Longer circuits → lower fidelity → higher RT cost → natural complexity penalty.

Protocol 3: Ancilla Budget Simulation

Simulate ancilla pool depletion:

```
class AncillaPool:
    def __init__(self, total_ancillas):
        self.remaining = total_ancillas

    def allocate_for_mint(self, num_ancillas):
        if self.remaining < num_ancillas:
            return None  # Minting impossible
        self.remaining -= num_ancillas
        return [Qubit(i) for i in range(num_ancillas)]

ancilla_pool = AncillaPool(total_ancillas=100)
for mint_attempt in range(max_attempts):
    ancillas = ancilla_pool.allocate_for_mint(5)
    if ancillas is None:
        break  # Pool exhausted → inflation stops
    bolt = mint_with_ancillas(ancillas)
```

Hybrid Classical-Quantum Architecture Since full quantum verification is resource-intensive, we propose a *two-tier verification system*:

1. **Light Classical Check (Fast):** Verify serial number format, timestamp, and RT ledger consistency. Computational cost: $O(\log M)$ where M is current money supply. Rejection rate: $\sim 90\%$ of invalid bolts (spam/malformed).
2. **Full Quantum Verification (Expensive):** Apply Zhandry's mini-verification protocol on quantum hardware. Only invoked for bolts passing classical checks. Computational cost: $O(n^3)$ quantum gates. Rejection rate: $\sim 100\%$ of remaining invalid bolts.

This *fast-path/slow-path* design reduces average verification cost by $10\times$, improving scalability.

Decentralized RT Ledger: Blockchain Integration The RT ledger must be:

- **Append-only:** Prevents retroactive RT forgery.
- **Consensus-driven:** Majority agreement on RT state prevents double-spending.

- **Lightweight:** Classical blockchain (e.g., Ethereum smart contract) suffices.

Smart Contract Pseudocode:

```
contract RTLedger {
    uint256 public totalRT;
    uint256 public remainingRT;
    mapping(address => uint256) public balances;

    function mintBolt(bytes32 serialNumber, uint256 rtCost)
        public returns (bool) {
        require(remainingRT >= rtCost, "Insufficient RT");
        remainingRT -= rtCost;
        emit BoltMinted(msg.sender, serialNumber, rtCost);
        return true;
    }
}
```

Temporal Ordering: Quantum-Classical Blockchain Hybrid We explore a *causally-ordered issuance* protocol where each bolt references its predecessor:

$$|\psi_n\rangle = U_n(\text{hash}(s_{n-1}))\,|\phi_{\text{seed}}\rangle\,,$$

where s_{n-1} is the previous bolt's serial number and U_n is a parameterized unitary.

Properties:

- **Chronological integrity:** Cannot mint bolt n without knowledge of bolt $n-1$.
- **Causal dependency:** Creates a directed acyclic graph (DAG) of monetary states.
- **Fork prevention:** Multiple concurrent bolts with same index n violate uniqueness.

This hybrid structure enables *quantum-verified timestamps* without requiring a full quantum blockchain.

Experimental Validation Plan We will validate physical feasibility through:

1. **Qiskit Simulation:** Run 100 minting attempts with RT tracking enabled. Measure: RT cost distribution, verification time, success rate.
2. **IBM Quantum Hardware (if available):** Deploy miniaturized circuit ($n = 2$, $m = 8$) on `ibm_brisbane` or similar. Compare: Ideal vs. noisy RT consumption, error correction overhead.
3. **Scalability Extrapolation:** Fit power-law model: $\text{RT}_{\text{cost}}(m) = am^b + c$. Extrapolate to production parameters ($n = 10$, $m = 200$). Estimate: Total RT budget needed for 1 million bolts.

Open Implementation Challenges Despite feasibility, several challenges remain:

- **RT Recharging:** Should consumed RT be permanently destroyed or recycled after bolt destruction? (Time-released cryptography may enable controlled recharging.)
- **Cross-Platform Standardization:** Different quantum hardware (superconducting vs. ion trap) have different gate costs—how to normalize RT pricing?
- **Adversarial Circuit Optimization:** Can attackers find circuit decompositions that artificially reduce measured RT cost while preserving functionality?

- **Quantum Memory Overhead:** Storing bolts as quantum states (vs. classical serial numbers) requires persistent quantum memory—currently unavailable.

We will discuss these challenges and propose mitigation strategies in the final report.

2.4 Comprehensive Validation Framework

Our validation methodology spans analytical derivation, numerical simulation, and experimental feasibility assessment.

Stage 1: Mathematical Foundation *Objective:* Derive closed-form or approximate solutions for monetary supply evolution under RT constraints.

Differential Equation Model: The rate of change of money supply $M(t)$ depends on minting rate $R(t)$ and destruction rate $\mu M(t)$ (due to lost/destroyed bolts):

$$\frac{dM}{dt} = R(t) - \mu M(t),$$

where the minting rate is bounded by:

$$R(t) = \min\left\{ \frac{Q(t)}{D(t)},\ \frac{\mathrm{RT}_{\mathrm{remain}}(t)}{\mathrm{RT}_{\mathrm{cost}} \cdot T_{\mathrm{mint}}} \right\}.$$

For exponential quantum growth $Q(t) = Q_0 e^{\lambda t}$, we analyze two regimes:
Regime I (RT-Unconstrained): When $\mathrm{RT}_{\mathrm{remain}} \gg 0$, minting is limited only by Q/D:

$$M_{\mathrm{unbound}}(t) \approx \frac{Q_0}{\lambda D} e^{\lambda t} \quad \Rightarrow \quad M \to \infty.$$

Regime II (RT-Constrained): When $\mathrm{RT}_{\mathrm{remain}}$ becomes limiting:

$$M_{\mathrm{bound}}(t) \to M_{\max} = \frac{R_{\max}}{\kappa m}\left(1 - e^{-\mu t}\right) \quad \Rightarrow \quad M \to \text{const}.$$

Equilibrium Analysis: Setting $dM/dt = 0$ yields equilibrium condition:

$$R_{\mathrm{eq}} = \mu M_{\mathrm{eq}} \quad \Rightarrow \quad M_{\mathrm{eq}} = \frac{R_{\max}}{\kappa m \mu}.$$

We will compute stability via linearization: $\delta M(t) = M(t) - M_{\mathrm{eq}}$ and show that eigenvalues of Jacobian have negative real parts (ensuring stable equilibrium).

Stage 2: Multi-Scenario Simulation *Objective:* Numerically verify theoretical predictions and explore parameter sensitivity.

Simulation Architecture:

1. **Classical Inflation Model (Baseline):** Implement Poisson process for minting: $P(\text{success}) = Q(t)/D$. Track cumulative supply $M(t) = \sum_{i=1}^{N(t)} 1$ over time.

2. **RT-Constrained Model (Proposed):** Augment with RT budget tracking: `if RT_remain < RT_cost: reject_mint()` After each successful mint: `RT_remain -= RT_cost`.

3. **Qiskit Circuit Simulation (Quantum):** Implement miniaturized Quantum Lightning circuits ($n = 3, m = 12$). Use three RT protocols (gate-count, decoherence, ancilla) in parallel. Measure actual RT consumption per bolt.

Experimental Parameters:

- Quantum growth rates: $\lambda \in \{0.1, 0.5, 1.0\}$ (conservative to aggressive)
- RT budgets: $R_{\max} \in \{10^3, 10^4, 10^5\}$ (scarcity levels)
- Difficulty schedules: constant D, linear $D(t) = D_0 + \alpha t$, adaptive $D(t) = f(R_{\text{obs}})$

Key Metrics to Extract:

1. **Inflation Rate:** $I(t) = \frac{1}{M(t)} \frac{dM}{dt}$ Target: $I_{\text{bounded}}/I_{\text{unbounded}} < 0.01$ (99% reduction).
2. **Stability Index:** $S = 1/\sigma_R$ where σ_R is variance of minting rate. Target: $S_{\text{bounded}} > 10 \times S_{\text{unbounded}}$ (more stable).
3. **Utilization Efficiency:** $\eta = M_{\text{actual}}/M_{\max}$ Target: $\eta > 0.9$ (minimal waste of RT budget).
4. **Time to Equilibrium:** t_{eq} such that $|M(t > t_{\text{eq}}) - M_{\text{eq}}| < 0.05 M_{\text{eq}}$. Target: $t_{\text{eq}} < 100 \times T_{\text{mint}}$ (reasonable convergence time).

Stage 3: Comparative Analysis and Visualization *Objective:* Produce publication-quality plots demonstrating the RT mechanism's efficacy.

Figure 1: Inflation Comparison Dual-axis plot showing:

- $M_{\text{unbound}}(t)$ (exponential curve, blue, right axis)
- $M_{\text{bound}}(t)$ (saturating curve, red, left axis)
- Crossover point where RT constraint activates (dashed vertical line)

Figure 2: RT Consumption Heatmap 2D heatmap: $(n, m) \to \text{RT}_{\text{cost}}$ measured from Qiskit simulations. Colormap intensity represents gate count; contour lines show iso-cost curves.

Figure 3: Verification Time Scaling Log-log plot: log(VerifyTime) vs. $\log(n)$. Fit to $O(n^3)$ polynomial (should show straight line with slope ≈ 3).

Figure 4: Stability Phase Diagram Phase space plot: $(\lambda, R_{\max}) \to$ Stability Regime. Color regions: "Stable" (green), "Marginal" (yellow), "Inflationary" (red).

Stage 4: Security and Attack Resistance Testing Beyond economic stability, we must verify the RT mechanism doesn't introduce vulnerabilities.

Attack Scenarios to Simulate:

1. **RT Exhaustion DoS:** Adversary mints bolts rapidly to deplete RT pool, preventing honest users from minting. *Mitigation:* Per-user rate limiting or proof-of-work for RT allocation.
2. **Circuit Optimization Exploit:** Adversary finds circuit decomposition that artificially lowers measured RT cost. *Mitigation:* Use hardware-intrinsic metrics (physical gate count) rather than logical gate count.
3. **Ledger Fork Attack:** Adversary creates conflicting RT ledger states to double-spend RT. *Mitigation:* Byzantine fault-tolerant consensus (Practical Byzantine Fault Tolerance or blockchain).
4. **Temporal Manipulation:** Adversary delays bolt announcement to game difficulty adjustment. *Mitigation:* Timestamp verification and network-wide clock synchronization.

For each attack, we will:

- Implement adversarial strategy in simulation
- Measure success probability and impact on $M(t)$

- Propose and test countermeasure effectiveness

Expected Deliverables for Part 2:

- Analytical closed-form solutions for $M(t)$ in both regimes
- Simulation codebase with three RT implementation variants
- Comprehensive plots (4 figures minimum) demonstrating efficacy
- Security analysis report identifying 4+ attack vectors and mitigations
- Parameter recommendation table: optimal $(n, k, m, \kappa, R_{\max})$ for target monetary policy

Part 3: Testing the Improved Model

Using the same simulation environment, we will reintroduce the improved model and test:

- Whether inflation stabilizes under increasing computational power.
- Whether the currency retains scarcity and uniqueness.
- Whether the protocol remains verifiable in polynomial time.

The simulation results will be analyzed for both theoretical safety and practical stability.

Part 4: Conclusion

We will summarize our findings and discuss whether the inflation-limited model can form a bridge between theoretical and physical realizations of quantum currency, addressing both security and economic stability.

3. Expected Outcomes

- A simulation demonstrating uncontrolled inflation in existing models.
- A modified issuance model that stabilizes monetary growth.
- Discussion of physical constraints linking computational complexity and currency supply.

4. Timeline and Sub-Milestones

Date	Milestone / Deliverable
Oct 30	Submit final proposal; finalize simulation framework selection (Python / Qiskit).
Nov 10	Implement base inflation simulation (Part 1). Validate unlimited inflation behavior.
Nov 17	Analyze Quantum Lightning model, identify inflation variables (Part 2.1).
Nov 24	Implement and test improved model (Part 2.2, 2.3).
Nov 30	Perform comparative simulation of inflation-limited vs. inflation-free models (Part 3).
Dec 5	Complete final report (Part 4) and prepare poster visualization.
Dec 9	Poster presentation and final defense of model results.

5. References

- Wiesner, S. "Conjugate Coding." *ACM SIGACT News*, 15(1), 78–88 (1983).
- Zhandry, M. "Quantum Lightning Never Strikes the Same State Twice." *arXiv:1711.02276v3*, 2019.
- Aaronson, S. *The Complexity of Quantum States and Transformations: From Quantum Money to Black Holes.* Bellairs Institute Lectures (2016).
- Nakamoto, S. "Bitcoin: A Peer-to-Peer Electronic Cash System." (2008).