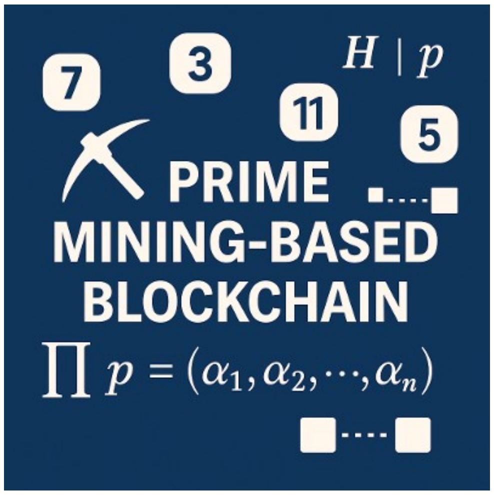
Midlincoln Research

December - 5 2025

Analyst: Semion Oganisian



Prime Mining as Cooperative Proof-of-Work: A Nashian Blockchain Based on Deterministic Prime Discovery

Abstract:

We present a blockchain architecture in which the fundamental proof-of-work consists of advancing the frontier of known primes in the integers. A block is valid only if it contains the next unused rational prime and certifies that all intervening integers are composite. Unlike hash-based proof-of-work, mining becomes a cooperative process: miners receive proportional rewards for contributing composite factorizations, while the prime discoverer receives the primary reward. This mechanism embodies Nashian cooperative equilibrium, ensuring that honest participation strictly dominates strategic deviation. Wallets maintain fractional ownership of each prime through a vector representation synchronized with an integer divisibility encoding. The resulting ledger features deterministic asset issuance, unique non-fungible prime assets, and immediate finality. We also describe optional extension chains defined over algebraic integer rings, enabling more complex forms of mathematical proof-of-work without altering the main chain.

Cryptography

1. Introduction

A decentralized digital currency requires a publicly verifiable mechanism for assigning value, recording ownership, and preventing double spending without trusted intermediaries. Bitcoin achieves this using a uniform currency unit and a competitive hash-based proof-of-work system in which miners race to solve useless puzzles. While effective, Bitcoin's design rewards *winner-takes-all* competition, concentrates mining power, and discards nearly all computational work.

This paper introduces a cryptocurrency built instead on **constructive number-theoretic work and Nashian cooperative mining incentives**. In this system, value arises from the mathematical structure of the prime numbers. Each prime—once discovered by proof-of-work—forms a unique and indivisible asset class. Instead of competing solely for a single reward, miners contribute collaboratively to certifying the integer interval leading to each newly discovered prime. Rewards are divided proportionally among contributors in ways that mirror **John Nash's principles of stable cooperative equilibria**: each participant receives a share commensurate with their demonstrable contribution, and no miner can improve their expected outcome by deviating from honest behavior.

Prime Assets and Ownership Encoding

The blockchain is an ordered sequence of rational primes. Mining consists of identifying the *next unmined prime* in ascending order. When a miner discovers prime p_{n+1}, it becomes a newly minted asset with total supply 1, initially assigned to the discovering miner. Wallets express ownership through a dual representation:

1. Headline Integer

A product of all primes in which the wallet holds a nonzero share.
If $p \mid H$, the wallet owns some portion of the asset p.

2. Fractional Ownership Vector

A vector $\{\alpha_p\}_{p \in P}$ with $\alpha_p \in [0,1]$, expressing the exact ownership fraction of each prime p.
The headline integer encodes *membership*, while the fractional vector encodes *magnitude*.

Transfers modify only the fractional coefficients, while headline integers update automatically to reflect whether the wallet's fractional balance of a given prime is zero or nonzero. Prime assets are therefore non-fungible, precisely indexed, and mathematically grounded: ownership is determined by simple divisibility, not arbitrary token identifiers.

Nashian Mining: Cooperative Proof-of-Work

Unlike Bitcoin’s purely competitive mining, this system introduces a **cooperative proof-of-work process**:

- Every number m between pn and pn+1 must be certified composite.
- Miners earn partial rewards for discovering and broadcasting valid composite factorizations.
- The prime-discovering miner receives the largest reward, but composite contributors also receive guaranteed shares.
- All work performed by miners is directly useful: composite proofs reduce verification cost, and primality proofs advance the chain.

This creates a **Nash-stable mining environment**:

- no miner gains by withholding composite proofs,
- no miner gains by deviating from honest interval scanning,
- rewards scale with demonstrable contribution,
- cooperative contribution dominates selfish strategies.

In contrast to classical PoW, where only one miner is paid and everyone else wastes computation, this system incentivizes broader participation and reduces centralization pressures.

Domain of Mining

The foundational blockchain operates in the rational integers Z.
This provides:

- deterministic prime ordering,
- guaranteed unique factorization,
- simple verification,
- and mathematically transparent scarcity.

The Z-chain acts as the stable monetary backbone.

Beyond this, the system allows **independent extension chains** in algebraic number rings such as Z[d] or the full ring of integers OK of a number field K.
In these domains miners identify irreducible algebraic integers ordered by increasing absolute norm, generating new asset classes with far more computationally demanding proof-of-work.
These extension chains operate entirely in parallel: they do not modify, fork, or otherwise influence the main Z chain.

Summary

This design yields a cryptocurrency grounded in mathematical scarcity, cooperative incentives, and transparent verification:

- **Prime-indexed asset classes** provide provable, permanent scarcity.
- **Fractional ownership vectors** define fine-grained value transfer.
- **Nashian cooperative mining** aligns incentives: work is rewarded proportionally, and honest participation is self-enforcing.
- **Integer-domain mining** ensures simplicity, determinism, and verifiability.
- **Optional algebraic extensions** offer room for advanced research without destabilizing the core ledger.

The result is a decentralized monetary system where mining corresponds not to arbitrary hash puzzles, but to constructive number-theoretic discovery—and where value arises from mathematics, cooperation, and equilibrium, not from competition alone.

2. System Overview and Data Structures

2.1 Ledger Structure

The system maintains a public, append-only ledger composed of blocks, each of which certifies a consecutive interval of integers and extends the frontier of known primes.
Each block contains:

1. **One newly discovered prime**
   A rational prime $p \in Z$ not previously mined.
   This prime becomes a new asset type with total supply 1.
2. **A sequence of transactions**
   These modify fractional ownership of previously mined primes through vector updates.
3. **Composite interval certifications**
   Valid factorization proofs for every integer between the previous prime pn and the new prime $pn+1$, enabling cooperative mining rewards.
4. **A state commitment**
   A Merkle root (or equivalent) summarizing the updated global prime list, all wallet vectors, and the resulting ledger state.

The ledger begins empty.
The first block contains prime 2, the next contains 3, then 5, 7, and so forth. Each prime appears **exactly once**, ensuring deterministic issuance, canonical asset creation, and resistance to replay or ambiguity.

The global ledger state consists of:

- an ordered list of mined primes

  $P=(p1,p2,\dots,pn)$,
- and a mapping from wallet identifiers to ownership vectors.

All nodes independently verify that a proposed pn+1 is the **unique smallest prime** larger than pn. No other value is acceptable, making issuance deterministic and preventing competing candidates.

This structure eliminates hash-based mining races:
there is only **one mathematically valid next block**.

2.2 Wallet Representation

A wallet's balance is expressed using two synchronized components, combining human readability with strict formal determinism.

(A) Headline Integer H(W)

For wallet W, define:

H(W)=p∈P:αp(W)>0∏p.

A prime p divides H(W) **if and only if** the wallet owns a positive fraction of asset p.

This provides a succinct integer summary of a wallet's *qualitative* membership across asset types.

(B) Fractional Ownership Vector α(W)

For each mined prime p∈P, a wallet stores:

αp(W)∈[0,1].

These coefficients specify *quantitative* ownership.

Whenever:

- a transfer reduces αp(W) $\rightarrow$ 0, p is removed from H(W);
- a transfer increases αp(W) from 0 $\rightarrow$ ϵ, p is inserted into H(W).

Thus:

- **divisibility** encodes *whether* a wallet owns a prime;
- **vector coefficients** encode *how much* it owns.

This dual encoding allows simple and deterministic validation based purely on integer arithmetic.

2.3 Global Prime Index and Deterministic Asset Order

The blockchain maintains the ordered list:

P=(p1,p2,…,pn),

where:

- p1=2,
- and for each i:

  and is primepi+1=min{p∈Z:p>pi and p is prime}.

Because the rational primes form a known sequence, this rule is:

- **unambiguous**,
- **publicly verifiable**,
- and **independent of mining strategy**.

This index determines:

- the structure and dimension of every wallet's vector,
- the order of asset introduction,
- and the canonical prime frontier for miners.

No miner can influence scarcity or choose which prime to mint.

2.4 State Commitments and Full Verification

Each block includes a commitment summarizing:

- the extended prime list P,
- all wallet headline integers H(W),
- all nonzero fractional balances αp(W),
- all applied transactions,
- and all composite proofs in the certified interval.

Nodes independently reconstruct the state upon receiving a block.

Verifying a Newly Mined Prime

To accept pn+1, a node checks:

1. **Primality**: the provided certificate verifies that pn+1 is prime.
2. **Uniqueness**: pn+1∈/P.
3. **Minimality**: there exists no prime p with pn<p<pn+1.
   Composite proofs for the entire gap provide this guarantee.

Verifying Transactions

Each transaction must satisfy:

1. The sender possesses sufficient fractional ownership.
2. Each updated coefficient remains within [0,1].
3. The headline integer is updated to reflect which primes have nonzero entries.
4. No fractional double spend occurs.

All checks involve only integer arithmetic and bounded rational updates—no floating-point operations, no probabilistic logic.

2.5 Persistence, Extensibility, and Nashian Structure

The integer prime blockchain functions as the **primary ledger**:
it is perpetual, deterministic, and globally stable.

Extension chains—operating in algebraic integer domains such as Z[d] or OK—may introduce:

- more complex irreducible elements as new asset types,
- higher proof-of-work difficulty,
- and richer number-theoretic structure.

However, the base Z-chain remains untouched and canonical.

This separation provides:

- **long-term stability** of the monetary layer,
- **backward compatibility**,
- **mathematical transparency**,
- and freedom for experimental, research-oriented chains.

Crucially, the mining process on the main chain incorporates **Nashian cooperative principles**:

- miners earn proportional rewards for composite interval certifications,
- the prime discoverer receives the largest share but benefits from others' work,
- no miner can unilaterally improve rewards by deviating from honest behavior,
- and the cooperative strategy profile becomes a stable equilibrium.

The result is a consensus mechanism where value creation aligns with mathematically meaningful work, and cooperative behavior is rational, stable, and economically rewarded.

3.2 Mining Rule: Discover the Next Prime

Given the current mined primes:

P=(p1,p2,…,pn),

the mining objective is to produce a block containing the next rational prime:

is primepn+1=min{p∈Z:p>pn, p is prime}.

A proposed block is valid **if and only if**:

1. Its prime is exactly pn+1,
2. Composite proofs certify all integers [pn+1,pn+1−1],
3. A deterministic primality certificate for pn+1 is included.

This mining rule ensures:

- **deterministic issuance**,
- **no ambiguity across miners**,
- **no competing candidates**,
- **no miner influence over scarcity**,

- and a **Nashian reward structure** where composite contributors and prime discoverers share block rewards proportionally to their verifiable effort.

3 — Proof-of-Work via Certified Prime Discovery

3.1 Overview

In this system, block production is tied to the discovery of the **next prime integer** after the chain's current frontier.
Proof-of-work consists of two components:

1. **Prime Discovery Proof** — a certificate that the newly submitted integer is prime and is the smallest prime greater than the previous block's prime.
2. **Composite Factorization Proofs** — short proofs that every integer between the last prime and the new prime is composite.

Mining therefore corresponds to *certifying* a continuous interval of integers:

is the previous block’s prime[n+1,pn+1]where pn is the previous block’s prime.

The chain only accepts the block whose interval is fully certified and ends at the true next prime.

3.2 The Frontier

Each block defines a **frontier**, the largest integer for which primality/compositeness is fully known by the network.

If block n contains prime pn, then the frontier is:

Fn=pn.

The next valid block must extend this coverage:

Fn+1=pn+1.

Thus, the blockchain maintains a certified classification of *all integers from 2 up to the current frontier*.

3.3 Mining Procedure

Miners operate independently on local computations. There is no central coordinator.

Starting from the frontier Fn:

1. For each integer m=Fn+1,Fn+2,…, miners attempt:
    - to find a **non-trivial factor** of m, and
    - to test m for primality.

2. If a miner finds a **valid composite proof**

   m=d·e,1<d<m,

   they broadcast a message:

composite_proof(m, d, e, miner_address)

All nodes verify composite proofs immediately (a single division check) and place them into their mempool.

- When a miner discovers the next prime pn+1, they assemble a block containing:
- the new prime pn+1,
- a **prime certificate** proving its primality, and
- **all composite proofs** for integers Fn+1 through pn+1−1, in strict order.
- The miner broadcasts:

1. block(p_{n+1}, prime_certificate, [composite_proofs], miner_address)

This concludes mining for that interval.

3.4 Composite Proofs

Composite proofs serve two roles:

1. **They make verification fast**:
   Nodes do not re-search the interval; they only check the factorization for each composite.
2. **They enable cooperative mining**:
   Multiple miners may contribute proofs and receive reward shares proportional to their contributions.

To remain Sybil-resistant, composite proofs are rewarded only when:

- they are the **first** valid proof for that integer m,
- they involve **non-trivial factorization** (configurable rule to avoid trivial spam, e.g., excluding divisors below a certain threshold).

3.5 Prime Certificates

A block must contain a deterministic and succinct primality certificate for pn+1.
Examples include:

- ECPP certificates,
- Pratt certificates,
- Pocklington’s test with proof,
- or any deterministic method verifiable in polynomial time.

Nodes verify the certificate without performing primality search themselves.

This maintains the crucial asymmetry of proof-of-work:

- **Mining is expensive**,

- **Verification is cheap**.

3.6 Block Validation Rule

Upon receiving a proposed block, a node performs:

1. **Frontier Consistency**
   The block must extend the known frontier exactly by classifying:

   Fn+1,Fn+2,…,Fn+1
2. **Composite Proof Verification**
   Each composite proof is checked via:

m % d == 0

f any integer in the interval lacks a proof, or proofs are out of order, the block is rejected.

1. **Prime Verification**
   The primality certificate for pn+1 must be valid.

If all checks pass, the block is accepted, and:

Fn+1=pn+1.

3.7 Network Communication Model (Gossip Layer)

The blockchain uses a **peer-to-peer gossip network** similar to Bitcoin:

- Each node maintains multiple TCP connections to peers.
- Nodes exchange binary messages (not HTTP).
- When a node receives:
  - a composite proof → it verifies and gossips it.
  - a block → it verifies and gossips it.

There is no central server.

Every node is both client and server at the TCP level, but no node has special authority. Consensus arises because all nodes independently execute the same deterministic validation rules.

3.8 Incentive Model

Block rewards are divided among contributors:

- **Prime finder** receives a fixed share Rprime.
- **Composite provers** collectively receive Rcomposite, split proportionally:

  reward(mi)=kRcomposite

  where k is the number of composite proofs in the block.

This enables a **cooperative PoW ecosystem** where resources are used efficiently and many miners earn partial rewards.

3.9 Security Considerations

- Composite proofs prevent denial-of-service attacks where verifying a large prime gap would otherwise require re-mining the interval.
- Succinct primality certificates prevent miners from presenting primes out of order.
- The gossip layer ensures rapid propagation of proofs and blocks.
- Reward fairness discourages centralization into giant pools.
- Deterministic validation prevents forks caused by ambiguous or probabilistic tests.

4 — Transactions and Fractional Ownership

4.1 Overview

In this blockchain, monetary value is represented by **primes**.
Each prime p corresponds to a unique asset type mined exactly once, at the moment it is discovered as the next block's prime.

Unlike classical cryptocurrencies where units are fungible, here:

- every prime has its own identity,
- ownership of each prime may be **fractional**,
- and total fractions of a prime always sum to exactly 1.

A wallet does not contain a scalar balance.
Instead, it contains a **prime-ownership vector**:

v=(c2,c3,c5,c7,…)

where each coefficient cp∈[0,1] expresses the fraction of prime p held by the wallet.

4.2 Prime Assets

When a new block discovers prime pn+1, that prime becomes a new asset in the system.
It is assigned:

- a **total supply of exactly 1 unit**,
- which initially belongs entirely to the block producer (or is split among miners based on the reward rules).

The prime becomes a tradable asset, independent of earlier primes.

Each prime resembles a *fully unique token* with fractional ownership, rather than a fungible currency unit.

4.3 Wallet Representation

Each wallet stores:

1. A **prime index list** (ordered list of all primes known so far).
2. A **vector of coefficients** matching that prime list.

For example:

Wallet A owns:

0 of 2 of 0.3 of 3,0.1 of 5,1.0 of 7.

Then its vector is:

vA=(c2=0,c3=0.3,c5=0.1,c7=1.0,c11=0,…)

Only primes that have been mined exist in the index.
Every wallet stores the same index but different coefficient vectors.

Another option is to hold a short vector vA=(c3=0.3,c5=0.1,c7=1.0) and reference the headline number H which is a product of all primes in the wallet with non zero coefficients order in ascending order.

4.4 The "Headline Integer"

Though all ownership is fundamentally expressed in the vector v, we may optionally define a **headline integer**:

H=p:cp=1∏p.

This expresses which primes the wallet owns **fully**, and is calculated easily from the vector.
It plays a philosophical role: it shows the “integer footprint” of the wallet, but is not needed for consensus.

4.5 Transactions

A transaction specifies:

- sender address
- receiver address
- set of primes involved
- fractional amounts to transfer
- digital signature

Example:
Wallet A wants to send **0.1 of prime 7** to Wallet B.

Transaction contains:

prime: 7

amount: 0.1

from: A

to: B

signature: sig(A)

Nodes verify:

- A’s vector has $c7 \geq 0.1$
- The signature matches A
- There is no double spending (fractions may not go negative)

After applying:

$c7A \leftarrow c7A - 0.1, c7B \leftarrow c7B + 0.1$

4.6 Fractional Transfers

Unlike Bitcoin, divisibility is not arbitrary: **fractions are bounded by [0,1] per prime**.
Thus fractional transfer rules are simple and deterministic.

Valid transfer must uphold:

- $0 \leq cpsender - amount \leq 1$
- $0 \leq cpreceiver + amount \leq 1$

No wallet can ever have >1 unit of a prime.

4.7 Wallet Integrity and Divisibility Logic

- If a wallet holds *any* fraction of a prime p, its coefficient vector reflects that.
- The vector, not the headline integer, determines true ownership.
- However another model can use the Headline number (a product of all primes in the wallet wiith non zero coefficient) and significantly shorter vector

Thus ownership is always unambiguous.

4.8 Transaction Fees

Each transaction pays a fee, expressed in **fractions of primes**.

Possible fee model:

- fee is paid in the smallest-index prime the sender owns a positive fraction of
- or fee is paid in any prime the sender chooses (more flexible)

Example:

- A holds 0.4 of prime 5

- Transaction fee is 0.01 of prime 5
- After sending, A has 0.39 of prime 5

Fees go to the block's miners according to the reward split.

4.9 Combining Fractional Vectors in a Block

A block includes:

- list of transactions
- modifications to each wallet’s vector
- composite proofs
- the new prime
- reward allocations

Nodes apply the vector updates in the order listed in the block.

Wallet vectors always remain valid because:

- all fractions are between 0 and 1
- total supply of each prime is preserved
- no two wallets can claim >1 cumulative unit of any prime

4.10 Minting New Prime Assets

The block’s prime discovery adds:

- a new prime pn+1
- a new coordinate to **every wallet's vector**

Initially:

- the miner who discovered new prime get 50% of reward gets cpn+1=0.5
- all other wallets whose addresses show proof for composites in the block split the rest of 50% and their coordinates for new prime are update proportionately to their contribution. Headline numbers are updated by multiplication by new prime.

This extends the vector dimension naturally.

Wallets do NOT need to rewrite history; they simply append 0 to their coefficient vector whenever a new prime is discovered.

4.11 Incentive Alignment

This system encourages:

- **miners**
  to supply composite proofs and find primes, earning new prime assets and fees.

- **users**
  to transact fractional ownership freely and safely.
- **validators**
  to check vector updates and prime certificates easily.

This creates a healthy, reward-aligned ecosystem.

4.12 Example Transaction Flow

1. Block N ends at prime 23.
2. Miners begin factoring 24–28 and searching for 29.
3. Block N+1 is found with prime 29.

In block N+1:

- Composite proofs for 24–28
- Primality proof for 29
- Rewards:
    - miner who found 29 gets 0.5 of prime 29
    - composite provers get fractional rewards of another 50% (or new prime depending on policy)

Now transactions:

- Wallet A sends 0.1 of prime 5 to Wallet B
- Wallet B sends 0.02 of prime 7 to Wallet C
- Fee of 0.01 of prime 3 is paid to miners

Wallet vectors update accordingly.

5 — Block Format and Network Message Types

The purpose of this chapter is to formalize the data structures used in the blockchain:

- how miners submit composite proofs,
- how blocks encode certified prime intervals,
- how transactions are packaged,
- how nodes gossip and request data,
- and how all fields are verified deterministically.

This creates a complete and self-contained protocol design.

5.1 Block Structure

Each block certifies a continuous interval of integers, extends the frontier to the next prime, and contains wallet state updates.

A block consists of:

```
Block {
   header: BlockHeader
   composites: [CompositeProof]
   transactions: [Transaction]
   rewards: RewardAllocation
}
```

Fields Explained

- **previous_block_hash**
  Links this block to the chain.
- **prime (p_{n+1})**
  The next prime in $\mathbb{Z}$.
- **prime_certificate**
  Deterministic proof of primality (e.g., ECPP or Pratt).
- **composite_range_start / composite_range_end**
  Defines the interval that must be certified as composite.
- **composite_merkle_root**
  Root hash summarizing all composite proofs.
- **transaction_merkle_root**
  Root hash summarizing all transactions.
- **miner_address**
  Wallet address receiving the prime reward share.

5.3 Composite Proofs

Each composite m requires exactly one proof:
a nontrivial factorization verifying:

$m = d \cdot e, 1 < d < m$

A composite proof structure:

```
CompositeProof {
   m
   factor: d
   provider_address
```

```
    signature
}
```

Nodes validate:

- $1 < d < m$
- m % d == 0
- signature corresponds to provider_address

Only the **first** valid proof for each m is accepted.

5.4 Transaction Format

Each transaction transfers fractions of prime assets between wallets.

```
Transaction {
    inputs: [ (prime, amount) ]
    sender_address
    receiver_address
    fee: (prime, amount)
    signature
}
```

Rules:

- The sender must hold enough fraction of each prime.
- Fractions must remain in [0,1].
- The fee must be small and paid in any prime the sender owns.

Transactions are bundled, hashed, and summarized by a Merkle tree.

5.5 Reward Allocation

Rewards are divided between:

1. **Prime finder** (major share, say 50%)
2. **Composite proof providers** (per composite) but they could split the remaining 50%

The block includes explicit reward entries

```
RewardAllocation {
    prime_finder_reward: (prime_(n+1), 0.5, miner_address)
    composite_rewards: [
        (prime_(n+1), amount (sums up to 0.5 in proportion to solving composits), provider_address),
        ...
    ]
}
```

Reward formats are identical to transactions, except they originate from the protocol, not a wallet.

5.6 Network Message Types (P2P Gossip Layer)

Nodes communicate using a simple peer-to-peer protocol over TCP.

Message Types:

version	— handshake message
verack	— handshake acknowledgment
inv	— announce hashes of blocks or proofs
getdata	— request full objects by hash
block	— deliver a full block
composite_proof	— deliver a composite factorization proof
transaction	— deliver a transaction
ping / pong	— keep-alives

5.7 Inventory Announcements (inv)

When a node receives:

- a new composite proof
- a new transaction
- a new block

it broadcasts an **inv** message to its peers:

```
inv {
    type: (block | composite | transaction)
    hash: object_hash
}
```

Peers request data they do not already have.

This is textbook gossip propagation.

5.8 getdata / block / composite_proof

When a node receives an inventory:

- If it does NOT have the object → it sends a **getdata** request.
- The peer responds with the full object.

Composite proof propagation example:

Peer A →

inv: composite_hash = abc123

Peer B →

getdata: composite_hash = abc123

Peer A →

composite_proof { m, d, address, signature }

5.9 Block Propagation

When a miner constructs a block, it broadcasts:

inv { type = block, hash = block_hash }

Peers then pull the block via getdata.

Nodes validate:

Composite verification

For each m:

check 1 < d < m

check m % d == 0

Prime verification

Validate primality certificate of p_(n+1).

Transaction verification

Check signatures and vector updates.

If valid → add to chain → gossip further.
If invalid → discard.

5.10 Initial Blockchain Download (IBD)

A new node:

1. Connects to peers.
2. Requests the chain of block headers.
3. Requests blocks in order.
4. Recomputes:
    - frontier
    - all wallet vectors
    - all composite intervals
    - entire transaction history

This mirrors Bitcoin’s design and ensures determinism.
No checkpoint servers needed.

5.11 Fork Resolution

When two blocks at the same height appear:

- Nodes accept the block whose prime is correct (the smaller next prime).
- Invalid chains are naturally rejected because block p_{n+1} must be the true next prime.
- There is no ambiguity: each integer has a unique primality status.

This gives stronger finality than hash-based PoW — the next prime is mathematically unique.

6 — Wallet State and Prime-Ownership Vector Algebra

Wallets in this system do not maintain a scalar currency balance.
Instead, each wallet stores a **prime-ownership vector** whose entries correspond to fractional ownership of each mined prime.

6.1 Prime Index and Vector Space

Let the sequence of primes discovered so far be:

$P=(2,3,5,7,\ldots,p_n)$.

Every wallet stores a vector:

$v=(c_2,c_3,c_5,c_7,\ldots,c_{p_n})$,

with constraints:

- $0 \le c_p \le 1$
- the **total supply** for each prime p satisfies:

  $\sum_{\text{all wallets}} c_p = 1$.

Thus the asset space is an **n-dimensional simplex**.

Another option wallet stores only coefficients for primes in Headline number ordered in ascending order.

6.2 Vector Updates

A transaction that transfers amount α of prime p from sender S to receiver R applies:

$cpS \leftarrow cpS - \alpha, cpR \leftarrow cpR + \alpha$.

All other coefficients remain unchanged.

Nodes reject a transaction if:

- $cpS < \alpha$, or
- $cpR + \alpha > 1$.

This ensures the invariants:

- No wallet overflows ownership of a prime.
- No prime is double-spent.
- All ownership is bounded and conserved.

6.3 Transaction Ordering and Determinism

Within a block, transactions are applied **in order of appearance**.
This guarantees deterministic vector evolution:

- given a block,
- and the previous state,
- all nodes compute exactly the same new vectors.

There are no floating-point operations; all fractional amounts use rational representations with fixed precision.

6.4 Adding a New Prime Coordinate

Whenever a new prime $pn+1$ is discovered:

- The prime is appended to the global prime index.
- Every wallet extends its vector with:

  $cpn+1=0$.
- Reward allocation sets:

  $cpn+1miner=1$.

This preserves:

- fixed ordering,

- consistent vector dimensionality for all wallets,
- total supply of 1 for each prime.

6.5 Headline Integer (Optional Convenience)

For any wallet with vector v, define:

$H(v)=\prod_{p\in P:c_p=1} p.$

This “headline integer” shows which primes a wallet owns fully but **does not affect consensus**. Ownership is entirely encoded in the vector.

The vector then doesn’t need to contain coefficients for all primes mined so far but only for those which divide the Headline number.

6.6 Wallet Synchronization

Nodes compute wallet vectors strictly from:

- the genesis state,
- all blocks,
- all transactions.

Thus:

- No wallet is authoritative.
- Private wallets may lose local state, but a full node can always reconstruct ownership.

7 — Incentive Structure and Economic Analysis

This chapter explains how the incentive model encourages:

- efficient resource use,
- decentralized participation,
- cooperation rather than competition,
- and Nash-stable behavior among miners.

7.1 Prime Discovery as Proof-of-Work

Finding the next prime p_{n+1} requires:

- scanning integers forward from the frontier,
- proving each integer composite or prime,

- producing the first valid primality certificate.

This requires computational work increasing with pn+1.
Rewards compensate miners for this cost.

7.2 Composite Proof Contributions

Miners who factor composite integers produce **composite proofs**:

- easy to verify (simple division)
- nontrivial to find
- economically valuable to the block creator

Composite proofs:

- reduce verification cost for all nodes,
- prevent block builders from re-mining the interval,
- distribute rewards fairly,
- allow smaller miners to participate profitably.

This creates a **multi-tiered mining economy**.

7.3 Reward Splitting

Total block reward R is split:

R=Rprime+Rcomposite+Rfees.

Where:

- Rprime: reward for discovering pn+1 - 50%
- Rcomposite: shared among composite proof contributors
- Rfees: accumulated transaction fees

  Rcomposite+Rfees=50%

For composite proofs:

reward(m)=#compositesRcomposite.

This model creates:

- a *big prize* (prime discovery),
- many *small prizes* (composite proofs),
- continuous income for miners.

7.4 Advantages over Hash-Based PoW

Compared to Bitcoin:

No wasted work

All computation (factoring, primality proving) advances mathematical knowledge and block validation.

Better decentralization

Small miners can profit from composite proofs.

Nash cooperative structure

Miners naturally specialize:

- Some test primality.
- Some factor composites.
- Some do both.

Equilibrium behavior is stable because:

- Rewards match measurable contributions.
- Composite proofs cannot be faked.
- Prime discovery is deterministic and unique.

7.5 Difficulty Adjustment Through Mathematics

Unlike Bitcoin, difficulty does not rely on probability.
It emerges from:

- the natural distribution of primes,
- the computational cost of primality proving,
- the cost of factoring integers in the interval.

As primes grow:

- gaps increase,
- primality certificates grow more expensive,
- composite factorizations become more difficult.

This yields a **natural difficulty curve** without explicit parameters.

7.6 Token Value and Market Behavior

Each prime is a unique asset class.
The value of prime p may correlate with:

- the computational cost of discovering p
- scarcity (later primes may be more valuable)
- use in transaction fees
- liquidity driven by user preference

This creates a **multi-asset economy** within one blockchain.

8 — Security Model

This chapter analyzes the system’s resistance to attacks:

- fork attacks
- message spamming
- composite proof manipulation
- censorship
- Sybil attacks
- malicious prime proposals

The design inherits Bitcoin’s decentralization while adding mathematical determinism.

8.1 Deterministic Prime Ordering Prevents Forks

Unlike Bitcoin, where chains can temporarily diverge, here:

- The next valid block must contain the *smallest* prime larger than the last.
- No miner can skip primes.
- No miner can propose an out-of-order prime without a valid certificate.

Thus:

is uniquely determined.pn+1 is uniquely determined.

Forks cannot be sustained.

8.2 Composite Proof Validation is Cheap

Composite proofs are validated by checking:

m≡0(modd).

This allows nodes to reject bad blocks instantly.

Even under a spam attack:

- invalid messages are cheap to detect,
- they propagate poorly,
- nodes ignore them immediately.

8.3 Preventing Composite-Proof Spam

To avoid trivial-factor spam:

- only the **first valid proof** for each m is accepted,
- trivial divisors (e.g., d ≤ threshold) may be excluded by policy,
- or only composite candidates passing a primality filter (Miller–Rabin) earn rewards.

This prevents gaming the reward system.

8.4 Censorship Resistance

A block builder cannot censor composite proofs:

- If they omit valid proofs, nodes will reject the block.
- If they include multiple proofs for the same m, nodes require the earliest one.

And because miners broadcast proofs immediately:

- censorship is impossible at network scale.

8.5 Sybil Attacks

Creating many fake identities does not help an attacker because:

- composite proofs are rewarded *once per integer*,
- not once per identity,
- fake nodes cannot forge prime or composite proofs,
- network connectivity does not grant extra rewards.

The system is naturally Sybil-resistant.

8.6 Invalid Prime Attack

A miner might try to propose:

- a composite number as the next prime, or
- a non-minimal prime (skipping primes).

Nodes reject such blocks because:

- primality certificates are verifiable,
- composite proofs for skipped m are missing.

Thus the attack is impossible.

8.7 Denial-of-Service on Nodes

Nodes validate blocks in:

- linear time for composite proofs,

- polynomial time for primality certificates.

Both are fast compared to mining.

Nodes also:

- discard malformed messages immediately,
- limit message rate per peer,
- ban peers sending invalid data.

The network remains robust under stress.

8.8 Finality

Because:

- primes are deterministic,
- certificate verification is deterministic,
- block intervals are deterministic,

finality is immediate after a single block.

No reorgs, no probabilistic confirmations.
A block either is the next prime block—or it is invalid.