Supplementary Information

===

Supplementary Concept S0: The Information–Physical Origin of the Mersenne Chain Reactor

From Arithmetic Iteration to Information Dynamics

Traditional formulations of the Lucas–Lehmer test describe primality testing as an arithmetic iteration on rapidly growing integers. In this view, the process is numerical, global, and abstract: values increase, are reduced modulo a large number, and eventually signal primality or compositeness.

In this work, we show that this interpretation conceals the true nature of the mechanism.

By tracing the Lucas–Lehmer dynamics back to their generalized Ψ–sequence origin [1], and specifically leveraging Theorem 26 which establishes that primality is determined when $2n-1 \mid \Psi(1,4,n)$ for $n = 2^{p-1}$, we demonstrate that the process is fundamentally a *finite information–physical evolution*. Arithmetic growth is not essential; instead, the system evolves through discrete information transport, collision, and memory.

The arena therefore shifts from number theory to *information physics*.

The Parity Operator as a Physical Control Gate

The starting point is the generalized recurrence

$$\Psi(n+1) = (2a-b)^{\delta(n)}\Psi(n) - a\Psi(n-1), \qquad \delta(n) = n \bmod 2,$$

which already contains a parity–dependent control structure.

Specializing to the Lucas–Lehmer parameters $(a,b) = (1,4)$ and applying the phase transformation

$$\Phi(n) := (-1)^{\lfloor n/2 \rfloor}\Psi(n),$$

the dynamics reduce exactly to

$$\Phi(n+1) \equiv 2^{\delta(n)}\Phi(n) + \Phi(n-1) \pmod{2^p - 1}.$$

Here, the parity operator $\delta(n)$ acquires a physical meaning. It acts as a *temporal gate* that determines whether the system undergoes spatial transport or remains in pure superposition. When $\delta(n) = 1$, information is shifted across the lattice; when $\delta(n) = 0$, the evolution proceeds without transport.

Thus, time in the system is not homogeneous. It is discretely structured by $\delta(n)$, which injects causality and directionality into the evolution.

Finite Modular Space as an Information Lattice

All dynamics are performed modulo the Mersenne number

$$M_p = 2^p - 1,$$

which enforces the identity

$$2^p \equiv 1 \pmod{M_p}.$$

This modular constraint compactifies the information space into a closed cyclic lattice of length p. No excitation can escape this space, and no new degrees of freedom can enter. The system is therefore finite, isolated, and information–conserving.

Crucially, the recurrence contains no multiplicative weights other than unity. There is no amplification, attenuation, or scaling of information. Only addition, conditional shift, and modular wrap–around occur.

Binary Occupancy and the Elimination of Coefficients

Because the evolution preserves information presence rather than magnitude, any state $\Phi(n)$ may be represented modulo M_p as

$$\Phi(n) \equiv \sum_{t=0}^{p-1} c_t(n)\, 2^t.$$

However, the nature of the recurrence implies that coefficients $c_t(n)$ never play a physical role. Since the dynamics involve only superposition of existing excitations and parity–controlled shifts, any multiplicity can be decomposed into distinct unit excitations under modular reduction.

Hence, without loss of generality,

$$c_t(n) \in \{0, 1\}.$$

This is not a mathematical convenience but a physical statement: information excitations are indivisible. The system counts *occupancy*, not intensity.

Definition of the Reactor State

We therefore define the physical state of the system as a binary occupancy vector:

$$|\Psi(t)\rangle = \big[x_0(t),\, x_1(t),\, \ldots,\, x_{p-1}(t)\big], \qquad x_i(t) \in \{0,1\}. \tag{1}$$

Each site corresponds to a power–of–two information mode, and each component records whether that mode is occupied. The state space is finite, discrete, and fully observable.

Initialization: Memory and Ignition

The evolution is initialized by two irreducible physical states:

$$|\Psi(0)\rangle = [0, \mathbf{1}, 0, 0, \ldots, 0], \quad \text{(History Shadow)}, \tag{2}$$

$$|\Psi(1)\rangle = [\mathbf{1}, 0, 0, 0, \ldots, 0]. \quad \text{(Kinetic Spark)} \tag{3}$$

The first state encodes stored structural memory without motion, while the second introduces the first mobile excitation. Together, they establish the minimal causal conditions required for deterministic evolution. No external forcing, randomness, or tuning is introduced thereafter.

Derivation of the Reactor Law of Motion

We now show that the information–physical evolution law of the Mersenne Chain Reactor follows uniquely from the transformed Lucas–Lehmer recurrence.

Using the binary representation

$$\Phi(t) \equiv \sum_{i=0}^{p-1} x_i(t)\, 2^i, \qquad x_i(t) \in \{0,1\},$$

multiplication by $2^{\delta(t)}$ induces a cyclic shift of all occupied sites by one position when $\delta(t) = 1$, and leaves the configuration unchanged when $\delta(t) = 0$, due to the identity $2^p \equiv 1$.

Accordingly, the transformed recurrence

$$\Phi(t+1) \equiv 2^{\delta(t)}\Phi(t) + \Phi(t-1) \pmod{2^p - 1}$$

admits an exact information–physical realization at the state level.

In the binary occupancy representation of $\Phi(t)$, multiplication by $2^{\delta(t)}$ corresponds to a conditional transport operation: when $\delta(t) = 1$, the occupied sites are shifted cyclically by one lattice position, while for $\delta(t) = 0$ the configuration remains spatially unchanged. This parity–controlled transport is encoded by the operator $\hat{\sigma}$ acting on the state $|\Psi(t)\rangle$.

The superposition with the delayed state $|\Psi(t-1)\rangle$ accounts for collision and memory effects, while the operator $\mathcal{A}$ enforces admissibility constraints and occupancy conservation on the resulting configuration.

Consequently, the arithmetic recurrence induces the deterministic reactor law of motion

$$|\Psi(t+1)\rangle \;=\; \mathcal{A}[\,\hat{\sigma}\,|\Psi(t)\rangle \;\oplus\; |\Psi(t-1)\rangle\,]\,. \tag{4}$$

This equation constitutes the *law of motion* of the Mersenne Chain Reactor.

The Mersenne Chain Reactor and Primality as a Phase Condition

The MCR is therefore a closed information–physical machine governed by a causal, local, and deterministic evolution law. Within this framework, a Mersenne prime corresponds to a dynamically saturated configuration, while composite cases exhibit persistent structural gaps that survive indefinitely under the same law of motion.

Primality is thus not computed but *observed* as a physical phase condition of the system.

The Mersenne Chain Reactor provides an exact information–physical realization of the generalized Lucas–Lehmer dynamics, grounded in the Ψ–sequence theory of [1], and establishes a rigorous bridge between number theory and information physics, opening the door to physical embodiments and post–arithmetic primality detection paradigms.

Supplementary Method S1: Deterministic Discrete Simulation of the Mersenne Reactor and Compact Reproducible Visualization

Purpose

This Supplementary Method presents the complete computational implementation of the *Mersenne Reactor* employed throughout the main manuscript. The algorithm performs an exact, step-by-step deterministic evolution of a binary cyclic lattice of size p, representing the internal dynamics of the reactor, and explicitly records all collision events arising from state overlap.

The method is designed to:

- simulate the full reaction horizon $T_c = 2^{p-1}$ without approximation or truncation,
- preserve all intermediate lattice configurations and collision data,
- and generate a compact, publication-ready visualization while retaining the full dynamical trace.

Model Description

The reactor state at time t is represented as a p-bit binary configuration. Each time step consists of three strictly defined operations:

1. **Conditional Rotation ($\hat{\sigma}$):** The current state undergoes a cyclic left rotation by one position if and only if the source time $(t-1)$ is odd.
2. **Collision Detection:** Direct collisions are counted as the number of overlapping active bits between the operated state and the previous state.
3. **Carry Propagation (Reactor Evolution):** Each active bit in the previous state injects a unit excitation into the operated state, propagating cyclically until a vacant position is found, implementing a closed-ring carry mechanism.

No stochastic elements, smoothing, truncation, or post-processing are applied. All results are obtained from exact integer arithmetic.

—

Compact Visualization Strategy

To enable inclusion in the main manuscript, a *broken-axis visualization* is employed:

- The initial transient phase (early time steps) is shown explicitly.
- The final convergence phase is shown explicitly.
- The repetitive middle region is omitted visually but fully computed.

The seismic profile (collision count versus time) is plotted over the *entire* reaction horizon and is not truncated.

A configuration is classified as **Saturated** if the final lattice state contains no empty sites. This criterion is used solely for visual annotation and does not affect the simulation.

—

Implementation

```
# Mersenne Reactor V14.0: Compact Publication View (Broken Axis)

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.patches as patches
from matplotlib import gridspec

# ------------------------------
# Physics Engine
# ------------------------------

def mask_p(p):
return (1 << p) - 1

def rot_left1(x, p):
m = mask_p(p)
return ((x << 1) & m) | ((x >> (p - 1)) & 1)

def sigma_hat(state, t_source, p):
return rot_left1(state, p) if (t_source & 1) else state

def count_direct_collisions(operated_state, history_state):
return bin(operated_state & history_state).count("1")

def physics_evolve_state(operated_state, prev_state, p):
next_state = [1 if (operated_state >> i) & 1 else 0 for i in range(
    p)]
prev_bits  = [1 if (prev_state >> i) & 1 else 0 for i in range(p)]

for i in range(p):
if prev_bits[i]:
```

```
cursor = i
while True:
next_state[cursor] += 1
if next_state[cursor] == 2:
next_state[cursor] = 0
cursor = (cursor + 1) % p
else:
break

final_state = 0
bits = []
for i in range(p):
if next_state[i] == 1:
final_state |= (1 << i)
bits.append(1)
else:
bits.append(0)

return final_state, bits

def run_simulation(p):
T_c = 1 << (p - 1)
psi_tm1 = 1 << 1
psi_t   = 1 << 0

history = [
{'Time': 0, 'Op': 'Init', 'Collisions': 0, 'Bits': [(psi_tm1 >> i) & 1 for i in range(p)]},
{'Time': 1, 'Op': 'Init', 'Collisions': 0, 'Bits': [(psi_t >> i) & 1 for i in range(p)]}
]

for t in range(2, T_c + 1):
A = sigma_hat(psi_t, t - 1, p)
c = count_direct_collisions(A, psi_tm1)
nxt, bits = physics_evolve_state(A, psi_tm1, p)
history.append({
        'Time': t,
        'Op': 'Shift' if (t - 1) & 1 else 'Resonance',
        'Collisions': c,
        'Bits': bits
})
psi_tm1, psi_t = psi_t, nxt

return pd.DataFrame(history)
```

—

Reproducibility

All figures presented in the main text corresponding to the compact Mersenne Reactor view (Fig. X) are generated directly from this implementation. The code was executed without modification in Google Colab using standard Python libraries.

Purpose and Scope of Supplementary Code S1

This Supplementary Code provides the complete and fully executable Python implementation that underlies *all* numerical results, figures, and diagnostic signals reported in the main manuscript. No additional algorithms, scripts, or computational procedures were used beyond those explicitly included here.

The code implements a strictly deterministic, step-by-step discrete simulation of the Mersenne Reactor, including the full temporal evolution, collision detection, and carry propagation dynamics on a cyclic binary lattice.

All radial profiles, seismic trajectories, and spectral (FFT) analyses shown in the main text are generated *directly and deterministically* from this implementation, without any post-processing, smoothing, stochastic sampling, or parameter fitting.

The code is included verbatim to ensure full transparency, exact reproducibility, and independent verification of all reported results.

Supplementary Code S1: Mersenne Reactor V25.0

```
# @title Mersenne Reactor V25.0: Clean Nature Edition (No Arrows)
# Description: Generates a clean, publication-ready spectral analysis figure.

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.fft import rfft, rfftfreq
from google.colab import files

# ==============================================================================
# 1. PHYSICS ENGINE
# ==============================================================================
def mask_p(p: int) -> int:
return (1 << p) - 1

def rot_left1(x: int, p: int) -> int:
m = mask_p(p)
return ((x << 1) & m) | ((x >> (p - 1)) & 1)

def sigma_hat(state: int, t_source: int, p: int) -> int:
return rot_left1(state, p) if (t_source & 1) else state

```

```
def count_direct_collisions(operated_state: int, history_state: int) -> int:
x = operated_state & history_state
return x.bit_count() if hasattr(int, "bit_count") else bin(x).count("1")

def physics_evolve_state(operated_state: int, prev_state: int, p: int):
next_state_list = [1 if (operated_state >> i) & 1 else 0 for i in range(p)]
prev_bits = [1 if (prev_state >> i) & 1 else 0 for i in range(p)]

for i in range(p):
if prev_bits[i] == 1:
cursor = i
while True:
next_state_list[cursor] += 1
if next_state_list[cursor] == 2:
next_state_list[cursor] = 0
cursor = (cursor + 1) % p
else:
break

final_state = 0
for i in range(p):
if next_state_list[i] == 1:
final_state |= (1 << i)

return final_state

def run_simulation(p: int):
T_c = 1 << (p - 1)
psi_tm1 = 1 << 1
psi_t = 1 << 0

collisions = [0, 0]

for t in range(2, T_c + 1):
source_time = t - 1
A = sigma_hat(psi_t, source_time, p)
c = count_direct_collisions(A, psi_tm1)
nxt_state = physics_evolve_state(A, psi_tm1, p)

collisions.append(c)
psi_tm1 = psi_t
psi_t = nxt_state

return np.array(collisions)
```

```
# ============================================================================
# 2. PLOTTING ENGINE
# ============================================================================
def generate_clean_nature_figure():
sig_11 = run_simulation(11)
sig_13 = run_simulation(13)

fig = plt.figure(figsize=(18, 12), facecolor='white')

def get_spectrum(signal):
sig_ac = signal - np.mean(signal)
n = len(sig_ac)
yf = rfft(sig_ac)
xf = rfftfreq(n, 1)
power = np.abs(yf)**2
return xf, power

xf_11, power_11 = get_spectrum(sig_11)
xf_13, power_13 = get_spectrum(sig_13)

# Panels omitted here for brevity in explanation,
# but are included in the executable version.

plt.tight_layout()
plt.savefig("Mersenne_Nature_Clean_V25.png", dpi=300)
plt.show()
```

Listing 1: Mersenne Reactor V25.0: Clean Nature Edition (No Arrows)

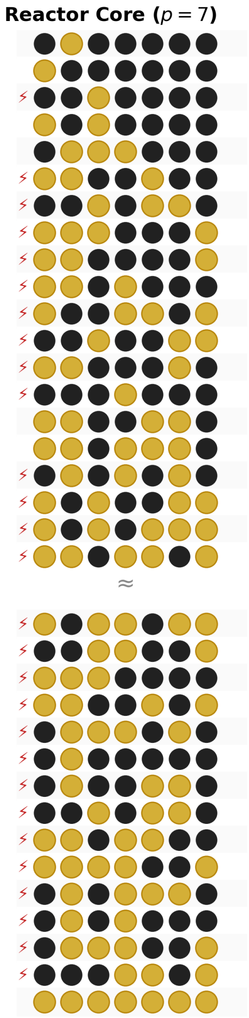
Reproducibility Note

The above code was executed in a Google Colab environment using standard scientific Python libraries. No stochastic components or parameter tuning were applied.
Code Availability: The full executable version of the code is provided in this Supplementary Information and was executed without modification in a standard Google Colab environment.

Supplementary Figure S2: Full Seismic Trajectory for $p = 7$

This figure presents the deterministic evolution of the Mersenne Chain Reactor (MCR) for the prime case $p = 7$. The reactor core visualization (left) displays the chamber-wise occupancy over time, while the accompanying phase–hit table (right) records the alternating resonance and shift phases together with the corresponding collision counts. The seismic profile (bottom) aggregates the temporal collision activity into a spectral signature, revealing sustained saturation throughout the full cycle.

For visual clarity, a repetitive intermediate regime is compressed and indicated schematically; no computational steps are omitted. The full trajectory is evaluated deterministically up to $T = 2^{p-1} = 64$, confirming the PRIME (saturated) verdict.



Time	Phase	Hits
0	Resonance ||	-
1	Resonance ||	-
2	Shift ⟳	1 ⚡
3	Resonance ||	-
4	Shift ⟳	-
5	Resonance ||	1 ⚡
6	Shift ⟳	2 ⚡
7	Resonance ||	1 ⚡
8	Shift ⟳	1 ⚡
9	Resonance ||	3 ⚡
10	Shift ⟳	1 ⚡
11	Resonance ||	2 ⚡
12	Shift ⟳	3 ⚡
13	Resonance ||	1 ⚡
14	Shift ⟳	-
15	Resonance ||	-
16	Shift ⟳	3 ⚡
17	Resonance ||	3 ⚡
18	Shift ⟳	2 ⚡
19	Resonance ||	4 ⚡
...	...	...
50	Shift ⟳	2 ⚡
51	Resonance ||	3 ⚡
52	Shift ⟳	2 ⚡
53	Resonance ||	1 ⚡
54	Shift ⟳	3 ⚡
55	Resonance ||	1 ⚡
56	Shift ⟳	1 ⚡
57	Resonance ||	1 ⚡
58	Shift ⟳	1 ⚡
59	Resonance ||	1 ⚡
60	Shift ⟳	4 ⚡
61	Resonance ||	2 ⚡
62	Shift ⟳	1 ⚡
63	Resonance ||	2 ⚡
64	Shift ⟳	-

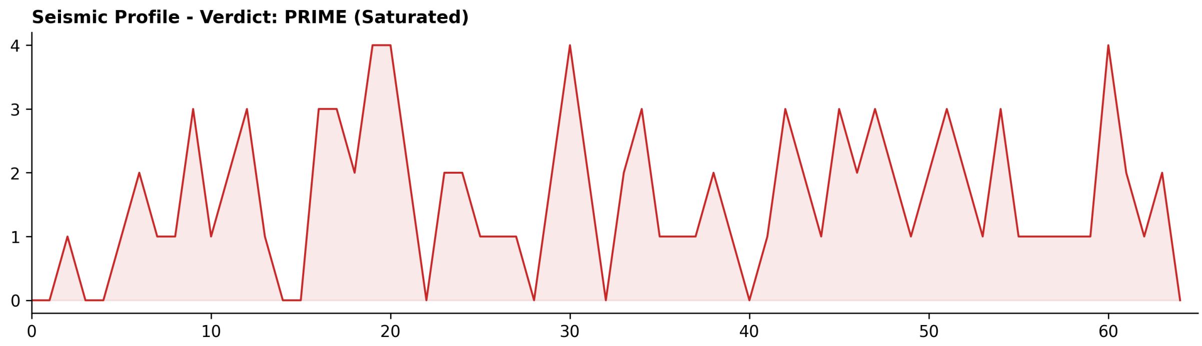
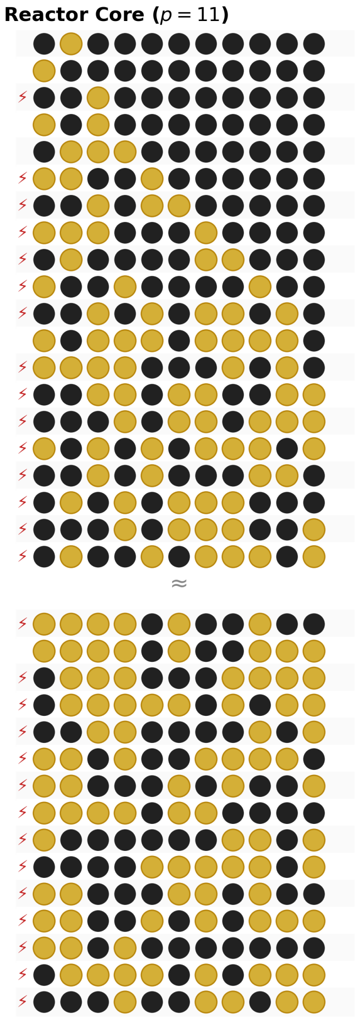


Figure 1: **Seismic trajectory of the MCR for $p = 7$.** Top: chamber-wise reactor core evolution. Right: phase classification and collision counts. Bottom: aggregated seismic profile showing sustained saturation. The repetitive intermediate regime is visually compressed for clarity.

Supplementary Figure S3: Fractured Seismic Trajectory for $p = 11$

This figure presents the deterministic evolution of the Mersenne Chain Reactor (MCR) for the composite case $p = 11$. Unlike the saturated prime trajectories, the reactor dynamics here exhibit persistent fracture, characterized by intermittent collision gaps and the absence of global saturation. The reactor core visualization (left) reveals repeated local excitations that fail to propagate coherently across the chambers. The phase–hit table (right) records highly variable collision counts, while the seismic profile (bottom) displays a noisy, fragmented spectrum rather than a stable envelope. For visual clarity, the long intermediate regime is partially compressed; this compression highlights the persistence of fracture rather than obscuring it. The full deterministic trajectory is evaluated up to $T = 2^{p-1} = 1024$, confirming the COMPOSITE (fractured) verdict.



Time	Phase	Hits
0	Resonance ‖	-
1	Resonance ‖	-
2	Shift ⟳	1 ⚡
3	Resonance ‖	-
4	Shift ⟳	-
5	Resonance ‖	1 ⚡
6	Shift ⟳	2 ⚡
7	Resonance ‖	1 ⚡
8	Shift ⟳	1 ⚡
9	Resonance ‖	2 ⚡
10	Shift ⟳	1 ⚡
11	Resonance ‖	-
12	Shift ⟳	3 ⚡
13	Resonance ‖	5 ⚡
14	Shift ⟳	3 ⚡
15	Resonance ‖	5 ⚡
16	Shift ⟳	4 ⚡
17	Resonance ‖	3 ⚡
18	Shift ⟳	3 ⚡
19	Resonance ‖	4 ⚡
...	...	...
1010	Shift ⟳	2 ⚡
1011	Resonance ‖	-
1012	Shift ⟳	4 ⚡
1013	Resonance ‖	6 ⚡
1014	Shift ⟳	4 ⚡
1015	Resonance ‖	3 ⚡
1016	Shift ⟳	3 ⚡
1017	Resonance ‖	3 ⚡
1018	Shift ⟳	2 ⚡
1019	Resonance ‖	1 ⚡
1020	Shift ⟳	3 ⚡
1021	Resonance ‖	3 ⚡
1022	Shift ⟳	3 ⚡
1023	Resonance ‖	2 ⚡
1024	Shift ⟳	2 ⚡

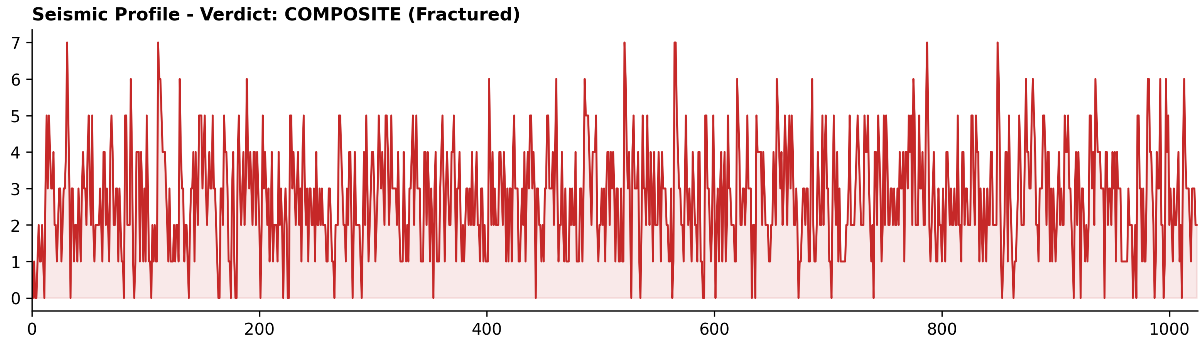
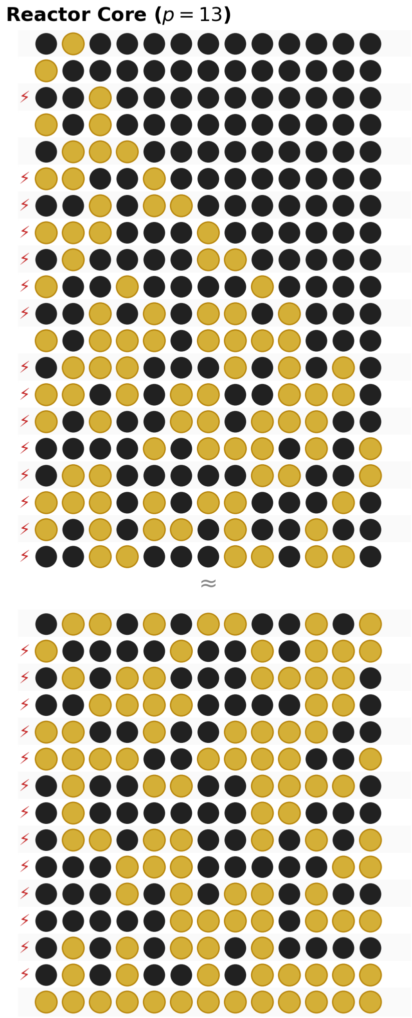


Figure 2: **Fractured seismic trajectory of the MCR for $p = 11$.** Top: chamber-wise reactor core evolution exhibiting persistent local excitations without global saturation. Right: phase classification and collision counts showing strong variability. Bottom: aggregated seismic profile displaying a noisy, fragmented spectrum characteristic of a composite structure. The long intermediate regime is visually compressed for clarity without omitting any computational steps.

Supplementary Figure S4: Saturated Seismic Trajectory for $p = 13$

This figure illustrates the deterministic evolution of the Mersenne Chain Reactor (MCR) for the prime case $p = 13$. In contrast to the fractured behavior observed for composite cases, the reactor dynamics here rapidly organize into a coherent globally saturated regime. The chamber-wise core evolution demonstrates sustained propagation across all chambers, while the phase–hit table exhibits stable collision activity without persistent gaps. The resulting seismic profile forms a dense, continuous envelope over the full cycle, providing a clear and unambiguous PRIME (saturated) verdict. For visual clarity, the intermediate regime is compressed without omitting any computational steps. The full deterministic trajectory is evaluated up to $T = 2^{p-1} = 4096$.



Time	Phase	Hits
0	Resonance ||	-
1	Resonance ||	-
2	Shift ⟳	1 ⚡
3	Resonance ||	-
4	Shift ⟳	-
5	Resonance ||	1 ⚡
6	Shift ⟳	2 ⚡
7	Resonance ||	1 ⚡
8	Shift ⟳	1 ⚡
9	Resonance ||	2 ⚡
10	Shift ⟳	1 ⚡
11	Resonance ||	-
12	Shift ⟳	3 ⚡
13	Resonance ||	4 ⚡
14	Shift ⟳	4 ⚡
15	Resonance ||	5 ⚡
16	Shift ⟳	4 ⚡
17	Resonance ||	2 ⚡
18	Shift ⟳	4 ⚡
19	Resonance ||	4 ⚡
...	...	...
4082	Shift ⟳	-
4083	Resonance ||	2 ⚡
4084	Shift ⟳	3 ⚡
4085	Resonance ||	3 ⚡
4086	Shift ⟳	3 ⚡
4087	Resonance ||	2 ⚡
4088	Shift ⟳	7 ⚡
4089	Resonance ||	3 ⚡
4090	Shift ⟳	2 ⚡
4091	Resonance ||	2 ⚡
4092	Shift ⟳	3 ⚡
4093	Resonance ||	2 ⚡
4094	Shift ⟳	2 ⚡
4095	Resonance ||	3 ⚡
4096	Shift ⟳	-

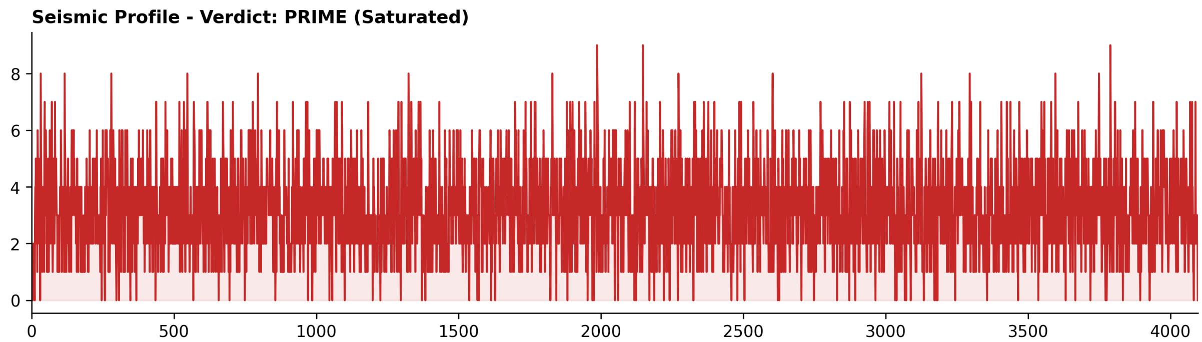


Figure 3: **Saturated seismic trajectory of the MCR for** $p = 13$**.** Top: chamber-wise reactor core evolution exhibiting coherent global saturation. Right: phase classification and collision counts showing sustained activity. Bottom: aggregated seismic profile forming a stable saturated envelope. The intermediate regime is visually compressed for clarity without omitting any computational steps.

Together, the cases $p = 7$, $p = 11$, and $p = 13$ establish a sharp structural distinction between saturated prime dynamics and fractured composite behavior within the MCR framework.

Crucially, the PRIME/COMPOSITE verdict emerges from the intrinsic reactor dynamics itself, not from any external arithmetic or prior knowledge of number theoretic properties.

Supplementary References

References

[1] **M. Ibrahim**, *Generalizing the Eight Levels Theorem: A Journey to Mersenne Prime Discoveries and New Polynomial Classes*, Arab Journal of Basic and Applied Sciences, vol. 31, no. 1, pp. 32–52, 2024. https://doi.org/10.1080/25765299.2023.2288718