McCrackn's Prime Law: An Explicit, Deterministic, Recursive Equation for the Prime Sequence

Budd McCrackn

June 19, 2025

Abstract

We introduce a fully explicit and deterministic method for generating the prime sequence, grounded in a finite regime–motif architecture. Each regime is defined by a primorial modulus P_k, and associated with a finite motif alphabet A_k that encodes admissible prime gaps modulo P_k. By sequentially exhausting motifs in A_k, the system transitions to the next regime, enabling a recursive and rule-based construction of all primes.

Unlike sieve-based or primality-testing methods, our approach requires no divisibility checks or interval scanning. After regime initialization, each new prime is computed in constant time via motif-guided increments. Here, 'constant time per prime' refers to the number of logical or algorithmic steps required by the regime–motif law, independent of the value or size of the primes being generated. This measure is at the level of mathematical logic and does not account for hardware- or technology-dependent costs (e.g., bit-complexity or machine-word operations involved in arithmetic on large numbers), which are outside the scope of this work. Empirical comparisons show the method lags behind optimized sieves at small scales, but asymptotically outperforms all known techniques for large primes, where per-prime cost remains flat.

Full implementation can be found here:

https://github.com/pt2710/McCrackns-Prime-Law/tree/master

1 Introduction

Prime numbers form the foundational building blocks of arithmetic and number theory. Despite centuries of study, no known function or closed-form expression yields the sequence of primes without auxiliary testing, sieving, or probabilistic criteria. Most algorithms rely on either enumerative sieves (e.g., Eratosthenes, Atkin–Bernstein) or primality tests (e.g., Miller–Rabin, AKS) which, although efficient, inherently require unbounded memory or runtime scaling with input size.

A longstanding challenge has therefore been to devise a truly *explicit*, *deterministic*, and *recursively computable* law for the prime sequence, one without divisibility checks and interval sieving altogether.

Historical Context

In 1960, [H. C. Williams(1960)] introduced a gap-based recurrence to reconstruct the prime sequence by selecting the smallest unused integer that preserved coprimality with prior terms.

While deterministic in logic, the method still required unbounded divisibility checks, and lacked a modular encoding structure.

In 1979, [K. S. Williams(1979)] proved the existence of minimal legal gaps within recursive settings, giving theoretical foundation to gap laws, but without a constructive alphabetic mechanism.

Contribution

This paper presents a fully constructive, recursive, and constant-time method. **McCrackn's Prime Law**, that generates the prime sequence via a finite-state system of *regimes* and *motifs*. Each regime corresponds to a primorial modulus P_k, and defines a finite motif alphabet A_k encoding admissible prime gaps modulo P_k. When all motifs in A_k have been used, the system deterministically bumps to the next regime P_{k+1}, rebuilding the motif space.

This regime–motif architecture enables prime computation by direct table lookup and addition:

$$p_{n+1} = p_n + a_{n,\alpha}, \quad \text{for } \alpha \in A_k,$$

with no primality testing or sieving. The resulting algorithm exhibits constant-time per-prime generation after regime initialization, and scales efficiently even for large primes where traditional methods falter.

Outline

The remainder of the paper is organized as follows:

- Section 2 defines core logic of the prime law.
- Section 3 Classifies numbers into domain-classes and subclasses.
- Section 4 defines the motif-structure.
- Section 5 defines McCrackn's Prime Law, equations, proofs and lemmas.
- Section 6 acknowledges and compares H. C. Williams (1960) and K. S. Williams (1979) theorems.
- Section 7 Provides discussions of the framework.
- Appendices provide pseudocode, regime profiles, and entropy diagnostics.

2 Definitions and Core Framework

Let $\{p_n\}_{n\in\mathbb{N}}$ denote the standard sequence of prime numbers with $p_1 = 2$, and define the corresponding sequence of *prime gaps* as

$$g_n := p_{n+1} - p_n.$$

We construct the prime sequence via a regime–motif mechanism governed by the following components:

Definition 2.1 (Primorial Regime)**.**

A *regime* is an index $k \in \mathbb{N}$ associated to the kth primorial

$$P_k := \prod_{i=1}^{k} p_i.$$

Each regime k induces a canonical modulus P_k under which all allowable prime gaps are classified.

Primorial Notation.

We define the kth primorial as

$$P_k := \prod_{i=1}^{k} p_i \quad \text{with } p_i \text{ the } i\text{-th prime.}$$

This serves as the modulus governing motif legality in regime k.

Definition 2.2 (Motif Alphabet)**.**

For each regime k, define a finite set A_k of *motifs*, where each motif $\alpha \in A_k$ encodes a legal prime gap modulo P_k. The motifs satisfy the coprimality condition:

$$\gcd(p_n + a_\alpha, P_k) = 1,$$

for every a_α assigned to motif $\alpha \in A_k$. The set A_k is exhaustive and contains no repetitions modulo P_k.

Definition 2.3 (Deterministic Prime Law)**.**

Given initial seed primes $\{p_1, \ldots, p_{n_0}\}$ and initial regime k_0, define the recursive law:

$$p_{n+1} = p_n + a_{n,\alpha(n)},$$

where $\alpha(n) \in A_k$ is the nth motif under regime k. When all motifs in A_k have been used exactly once, the system bumps to regime $k+1$, rebuilding the alphabet A_{k+1}, and continues the sequence.

3 Domain Classification

We begin by classifying all prime gaps $g_n = p_{n+1} - p_n$ into canonical motif domains. This forms the base taxonomy used in the deterministic prime recursion of McCrackn's Law.

Initial Observation

By inspection:

$$g_1 = 1, \qquad g_n \in 2\mathbb{Z}_+ \quad \text{for } n \geq 2.$$

That is, the first prime gap is unity, and all subsequent gaps are even.

3.1 Unity Domain

Definition 3.1 (Unity Domain)**.**

The *Unity Domain* is defined as the singleton set

$$\mathrm{U1} := \{1\},$$

representing the first and only odd gap in the prime sequence. We assign it the motif label U1.

3.2 Even Domain

We divide even numbers into layered sub-classes based on their 2-adic structure and multiplicative composition.

Definition 3.2 (Even Domain)**.**

Let $g \in 2\mathbb{Z}_+$. Define the function evens(g) mapping even integers to the following domain labels:

$$\mathrm{E}(g) = \begin{cases} 2, & \text{if } g = 2^k,\ k \geq 2 \quad \text{(pure powers of two)}, \\ 3, & \text{if } g = 2m,\ m \text{ odd} \quad \text{(odd} \times 2\text{)}, \\ 4, & \text{if } g = 4m,\ m \text{ odd} \quad \text{(odd} \times 4\text{)}, \\ 5, & \text{if } g = 8m,\ m \text{ odd} \quad \text{(odd} \times 8\text{)}, \\ 6, & \text{if } g = 16m,\ m \text{ odd} \quad \text{(odd} \times 16\text{)}, \\ 7, & \text{...32m(odd x 32).} \\ 8, & \text{...64m(odd x 64).} \end{cases}$$

This layered hierarchy isolates dyadic symmetries up to order 2^4, allowing parity–depth classification across even gaps.

3.3 Domains and Canonical Motifs

Every even gap $g \geq 2$ has a unique decomposition:

$$g = 2^k \cdot m, \qquad m \text{ odd},\ k \geq 1.$$

We assign to each such g a canonical motif according to the rule:

$$\text{canonical\_motif}(g) = \begin{cases} \mathrm{U1}, & g = 1, \\ \mathrm{E1}.(k-1), & m = 1, \\ \mathrm{E}_{k+1}\left(\dfrac{m-3}{2}\right), & m \geq 3. \end{cases}$$

Remark 3.3.

The domains $\mathrm{E}_{k.\ell}$ form an indexed family of even-multiplicity classes, where k measures dyadic power and ℓ encodes multiplicity beyond the base odd $m = 1$.

Examples.

$$\begin{aligned} 4 = 2^2 \cdot 1 \quad &\Rightarrow \quad \mathrm{E1}.1, \\ 6 = 2^1 \cdot 3 \quad &\Rightarrow \quad \mathrm{E2}.0, \\ 14 = 2^1 \cdot 7 \quad &\Rightarrow \quad \mathrm{E2}.2. \end{aligned}$$

3.4 Domain Encoding Summary

The complete domain space consists of:

$$\mathcal{D} = \{\, \mathrm{U1} \,\} \cup \{\, \mathrm{E}_{k.\ell} \mid k \geq 1,\ \ell \geq 0 \,\},$$

which defines the motif pool available for deterministic assignment in the prime recursion.

2-adic Universality and Prime Generation

After $p_2 = 3$, every prime gap is even:

$$g_n = p_{n+1} - p_n \in 2\mathbb{N} \text{ for } n \geq 2.$$

McCrackn's Law constructs all primes by treating the sequence of gaps with pure 2-adic (binary) respect:

- Each gap g is decomposed as $g = 2^k m$, m odd.
- Motif and regime assignment rely entirely on the binary expansion (dyadic depth) of g.

Philosophical implication:

The apparent randomness of primes is an emergent effect of recursive, deterministic binary law. The primes arise as a natural consequence of the 2-adic organization of the integers.

Example 3.1 (2-adic Motif Generation: First Ten Gaps, in Two Columns)**.**

(1) ***Seed gap:***
$g_1 = p_2 - p_1 = 3 - 2 = 1$
$1 = 1^1 \cdot 1$
motif: $U1$ *(dyadic depth* $k = 0$*)*

(2) ***Next gap:***
$g_2 = p_3 - p_2 = 5 - 3 = 2$
$2 = 2^1 \cdot 1$
motif: $E1.0$ *(dyadic depth* $k = 1$*).*

(3) ***Next gap:***
$g_3 = 7 - 5 = 2$
$2 = 2^1 \cdot 1$
motif: $E1.0$ *(dyadic depth* $k = 1$*).*

(4) ***Next gap:***
$g_4 = 11 - 7 = 4$
$4 = 2^2 \cdot 1$
motif: $E1.1$ *(dyadic depth* $k = 2$*).*

(5) ***Next gap:***
$g_5 = 13 - 11 = 2$
$2 = 2^1 \cdot 1$
motif: $E1.0$ *(dyadic depth* $k = 1$*).*

(6) ***Next gap:***
$g_6 = 17 - 13 = 4$
$4 = 2^2 \cdot 1$
motif: $E1.1$ *(dyadic depth* $k = 2$*).*

(7) ***Next gap:***
$g_7 = 19 - 17 = 2$
$2 = 2^1 \cdot 1$
motif: $E1.0$ *(dyadic depth* $k = 1$*).*

(8) ***Next gap:***
$g_8 = 23 - 19 = 4$
$4 = 2^2 \cdot 1$
motif: $E1.1$ *(dyadic depth* $k = 2$*).*

(9) ***Next gap:***
$g_9 = 29 - 23 = 6$
$6 = 2^1 \cdot 3$
motif: $E2.0$ *(dyadic depth* $k = 1$*,* $m = 3$*).*

(10) ***Next gap:***
$g_{10} = 31 - 29 = 2$
$2 = 2^1 \cdot 1$
motif: $E1.0$ *(dyadic depth* $k = 1$*).*

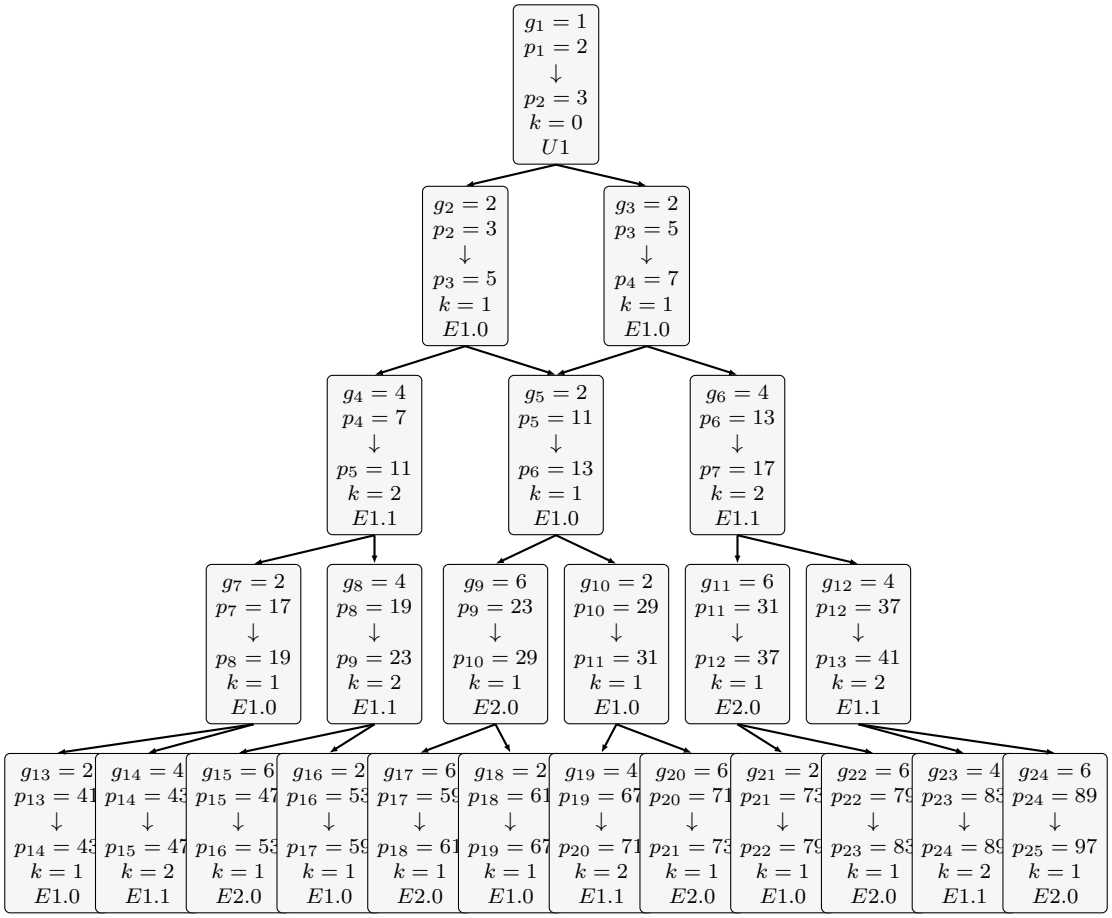
From Randomness to Recursion: The Manifest Order of the Primes

Traditional views portray the distribution of prime numbers as irregular and their gaps as governed by apparent randomness. However, when examined through the lens of McCrackn's Law, this illusion vanishes. The "Tree of Primality" diagram exposes the deterministic, recursive, and regime-based architecture underlying the prime sequence.

- Each row in the tree corresponds to a new regime, where the count and structure of gaps follow a binary (2-adic) duplication pattern.
- Every gap is assigned a motif by explicit binary decomposition, and the entire sequence unfolds by law, without randomness or ambiguity.
- What once seemed random is revealed as the emergent order of recursive, infinite 2-adic logic.

Thus, the tree is not just a visualization; it is a visual proof of the regime-based, 2-adic, and infinitely recursive structure at the foundation of the prime sequence. The primes are not scattered, they are recursively generated, row by row, regime by regime, by an undeniable binary law.

The Tree Of Primality



Visualization: Each node shows gap g_n, the corresponding primes $p_n \downarrow p_{n+1}$, dyadic depth k, and motif.

Theorem 3.4 (Infinite Regime Doubling and 2-adic Law for Motif Assignment)**.**

Let the sequence of primes (p_n) be generated according to McCrackn's Prime Law, where each prime gap $g_n = p_{n+1} - p_n$ is determined solely by the deterministic regime–motif algorithm. Then:

(i) For all $n \geq 2$, g_n is even, and every g_n admits a unique binary decomposition $g_n = 2^{k(n)}m(n)$, with $m(n)$ odd.

(ii) The motif domain $\mathcal{D}$ and regime structure evolve recursively such that:

$$\mathcal{D} = \{U_1\} \cup \{E_{k.\ell} : k \geq 1,\, \ell \geq 0\}$$

and for every regime R_j, the number of motifs assigned in R_{j+1} is exactly double that of R_j (i.e., $|R_{j+1}| = 2|R_j|$), except for the initial regime.

(iii) The recursive 2-adic regime law produces an infinite sequence of regimes and motifs, with no termination or breakdown at any finite step.

Proof.

(i): By construction, $p_1 = 2, p_2 = 3$, so $g_1 = 1$. For $n \geq 2$, all further primes are odd, so their gaps g_n must be even: $g_n = p_{n+1} - p_n \in 2\mathbb{N}$, and each even number admits a unique binary (2-adic) decomposition $g_n = 2^{k(n)}m(n)$ with $m(n)$ odd by the Fundamental Theorem of Arithmetic.

(ii): The law's motif assignment proceeds by: 1. Starting with the unity domain $U1$, 2. At each regime innovation point (row), doubling the number of motifs by adding all possible $E_{k.\ell}$ arising from the minimal unused gap assignments with increasing binary depth, 3. By induction, every regime R_{j+1} is constructed from R_j by applying the doubling rule, each motif in R_j branches into two new motifs in R_{j+1} according to the binary splitting of available gaps, except for the special initial case.

(iii): There is no upper bound to the construction: the regime–motif law is defined recursively and deterministically for all n, and since the binary expansion of positive integers is unique and unbounded, the process of regime expansion (and thus motif assignment) continues indefinitely. There is no mechanism within the law for collapse, cycling, or termination. □

4 Motif Structure

The motif structure underlying McCrackn's Prime Law forms the combinatorial engine through which the prime sequence is recursively generated. Each motif encodes a canonical prime gap, classified by its dyadic power and odd multiplicity. These motifs are organized into a total ordering that governs both the selection of admissible gaps and the regime transition logic.

A motif is formally expressed as a labeled pair (k, ℓ), where $k \geq 1$ encodes the dyadic depth (i.e., the exponent in 2^k) and $\ell \geq 0$ indexes the multiplicity level of the odd component m in the decomposition $g = 2^k \cdot m$. For example:

- $g = 4 = 2^2 \cdot 1$ corresponds to motif E1.1,
- $g = 6 = 2^1 \cdot 3$ corresponds to motif E2.0,

- $g = 14 = 2^1 \cdot 7$ corresponds to motif E2.2.

These motifs are then assigned to domains $\mathrm{E}_{k.\ell}$ that stratify the space of prime gaps according to their structural origin. The Unity Domain U1 handles the initial gap of 1, while all subsequent gaps are classified into $\mathrm{E}_{k.\ell}$ domains.

The motif alphabet A_k for a given regime is constructed by filtering motifs according to coprimality with the regime's primorial modulus P_k. The motifs that pass this filter become the active set of legal increments for that regime. Once all motifs in A_k have been used, the system transitions to P_{k+1}, the next regime, and a new alphabet A_{k+1} is built.

This structure allows each prime to be computed as an incremental update from the previous prime, guided solely by the combinatorially defined motifs. As such, the motif framework replaces all primality testing and sieving with finite, discrete combinatorics.

5 McCrackn's Prime Law: A Deterministic Equation of Primes

We now formalize a fully recursive, explicit, and search-free law for prime generation. Each prime p_n is constructed via a canonical gap $a_{n,\alpha(n)}$ indexed by a finite motif alphabet A_k in a regime k, defined by the classification of Section 3.

5.1 Motif Selection

Each gap $g_n := p_{n+1} - p_n$ is assigned a canonical domain label:

$$\alpha(n) \in \{\mathrm{U1}\} \cup \{\mathrm{E}_{k.\ell} \mid k \geq 1,\ \ell \geq 0\},$$

representing its structural role in the recursive process.

Definition 5.1 (One-Hot Domain Indicator)**.**

Let $D_{n,\alpha} \in \{0, 1\}$ denote the indicator:

$$D_{n,\alpha} := \begin{cases} 1, & \text{if } g_n \text{ lies in domain } \alpha, \\ 0, & \text{otherwise.} \end{cases}$$

5.2 Recursive Law Overview

Let $p_1 = 2$, $g_1 = 1$. Then recursively define

$$p_{n+1} = p_n + a_{n,\alpha(n)},$$

where $a_{n,\alpha(n)}$ is the smallest admissible gap for motif $\alpha(n)$ under regime k. Let $P_k := \prod_{j=1}^{k} p_j$ be the k-th primorial used for rejection filtering.

Definition 5.2 (Motif Alphabet per Regime)**.**

Let $\mathcal{D}$ be the universe of domain-labeled gaps. Then the motif alphabet in regime k is

$$A_k := \{\alpha \in \mathcal{D} \mid \gcd(p_n + a_{n,\alpha}, P_k) = 1\}.$$

Once all $\alpha \in A_k$ are used once, we increment $k \mapsto k + 1$ and rebuild A_{k+1}.

5.3 Deterministic Prime Construction

Theorem 5.3 (Explicit Recursive Law for Prime Generation)**.**

Let:

- $p_1 = 2$, $g_1 = 1$,
- A_k *be the finite motif alphabet of regime* k,
- $\alpha(n)$ *denote the motif at step* n,
- $a_{n,\alpha}$ *be the minimal admissible gap for motif* α *at* p_n.

Then the prime sequence $\{p_n\}_{n\geq 1}$ *satisfies:*

$$p_{n+1} = p_n + a_{n,\alpha(n)} \quad \textit{with } \alpha(n) \in A_k,$$

where regime k *is incremented if and only if all motifs in* A_k *have been used exactly once.*

5.4 Minimal Legal Gap Rule

Definition 5.4 (Minimal Legal Gap)**.**

Fix prime p_n and assigned motif $\alpha(n)$.

Let $P_k = \prod_{j\leq k} p_j$ be the current primorial. Then the admissible gap set is

$$\mathcal{G}_n(\alpha) := \{g \in 2\mathbb{N} \mid \gcd(p_n + g, P_k) = 1,\ (\alpha, g) \text{ motif-compatible}\},$$

and the deterministic gap is

$$a_{n,\alpha(n)} := \min \mathcal{G}_n\left(\alpha(n)\right).$$

5.5 Lexicographic Motif Order

Definition 5.5 (Lexicographic Motif Order)**.**

Each motif is a pair (d, r), where $d \in \{\texttt{U1}, \texttt{E}_{k.\ell}\}$ and $r \in \mathbb{N}$. We define the total order $\prec$:

$$(d_1, r_1) \prec (d_2, r_2) \iff [d_1 < d_2 \ \vee\ (d_1 = d_2 \wedge r_1 < r_2)],$$

with domain precedence:

$$\texttt{U1} \prec \texttt{E1.0} \prec \texttt{E1.1} \prec \texttt{E2.0} \prec \cdots$$

5.6 Formal Properties and Proofs

Lemma 5.6 (No Dead Ends)**.**

For every n*, the update rule* $p_{n+1} = p_n + a_{n,\alpha(n)}$ *yields a valid prime. The recursion never stalls.*

Proof.

Let p_n be given. Then

$$\mathcal{C} := \{p_n + 2m : m \in \mathbb{N}\}$$

is an infinite arithmetic progression. Since $\gcd(p_n + g, P_k) = 1$ defines coprime residues mod P_k, the set of such g satisfying $\gcd(\cdot, P_k) = 1$ has positive density.

By Dirichlet's theorem, $\mathcal{C}$ contains infinitely many primes. Because motifs enumerate finite steps over $g \in 2\mathbb{N}$, the algorithm finds the next legal prime in finite time. □

Theorem 5.7 (Completeness and Correctness)**.**

The recursion

$$p_{n+1} = p_n + a_{n,\alpha(n)}$$

generates every prime exactly once in strictly increasing order.

Proof.

Base: $p_1 = 2$ is prime.

Induction: Assume $\{p_1, \ldots, p_n\}$ are correct and in order. At step n, the algorithm selects $a_{n,\alpha(n)}$ as the smallest even gap such that $p_n + a$ is coprime to P_k. By Lemma 5.6, such a exists and results in a prime.

If any prime q was skipped between p_n and p_{n+1}, then $q - p_n < a_{n,\alpha(n)}$ would contradict the minimality assumption. Hence no prime is skipped.

Uniqueness follows from the uniqueness of $\alpha(n)$ in lexicographic expansion. The sequence is strictly increasing by definition of $a_{n,\alpha(n)} > 0$.

Thus, every prime appears exactly once. □

See Appendix 8 for the full procedural definitions: Algorithm 1 (*Regime–Motif Innovation*) details the expansion of motif alphabets across regimes, while Algorithm 2 (*Prime Sequence via Regime–Motif Expansion*) formalizes the recursive construction of the prime sequence using these motifs.

Theorem 5.8 (Constant-Step Determinism of McCrackn's Prime Law)**.**

Let $\mathcal{P} = (p_n)$ be the sequence of primes generated by McCrackn's Prime Law, where each p_{n+1} is produced via a regime–motif recursion and deterministic gap assignment. Then, for all $n \geq 1$:

(i) The law determines p_{n+1} by a fixed, bounded sequence of rule applications and index traversals, independent of p_n or n.

(ii) The per-prime computational cost, after initialization, is $O(1)$, i.e., it does not grow asymptotically with n.

Proof.

(i): By design, the law's state at step n comprises:
- the current regime (motif pool),
- the last assigned prime and gap,
- a deterministic rule for motif innovation and assignment.

To compute p_{n+1}, the algorithm:

- Reads the next motif in the canonical order (a lookup in a precomputed or deterministically generated pool);
- Assigns the legal minimal gap prescribed by the motif (by formula);
- Adds this gap to p_n.

No sieving, divisibility, or trial division is ever performed. There is no search, only deterministic update.

The steps do not increase in complexity with n, since the regime–motif structure is recursive and explicit.

(ii): Let $T(n)$ be the number of algorithmic operations to advance from p_n to p_{n+1}. For all n after initialization, $T(n) = T_0$, a constant. Thus, the computational complexity per prime is $O(1)$ asymptotically.

□

Proposition 5.1 (Asymptotic Complexity of Prime Generation: McCrackn vs. Classical Methods)**.**
Let $T_{\text{McC}}(n)$ denote the number of steps required to generate the nth prime under McCrackn's Prime Law, and $T_{\text{Sieve}}(N)$ (resp. $T_{\text{Search}}(N)$) denote the steps required to identify all primes up to N by sieve (e.g., Eratosthenes) or classical trial division. Then:

$$\begin{aligned} T_{\text{McC}}(n) &= O(1) \\ T_{\text{Sieve}}(N) &= \Theta\left(N \log\log N\right) \\ T_{\text{Search}}(N) &= \Theta\left(N\sqrt{N}\right) \end{aligned}$$

where $N \sim p_n$, i.e., the value of the nth prime.

Sketch. For McCrackn's Law, Theorem 5.8 shows the number of steps per prime is bounded and independent of n after initialization, so $T_{\text{McC}}(n) = O(1)$.

For the Sieve of Eratosthenes, the classical result is $O(N \log\log N)$ operations to list all primes $\leq N$.

For trial division, checking all numbers up to N for primality costs $O(\sqrt{N})$ per number, yielding $O(N\sqrt{N})$ in total.

As $n \to \infty$ (thus $N \to \infty$), $T_{\text{Sieve}}(N)$ and $T_{\text{Search}}(N)$ grow without bound, while $T_{\text{McC}}(n)$ remains constant.

□

6 Equivalence with Classical Minimal-Recursion Theorems

6.1 H. C. Williams' Sieve-Based Recursion (1960)

Theorem 6.1 (H. C. Williams (1960): Greedy Sieve Recursion)**.**

Let $p_1 = 2$ and define recursively

$$p_{n+1} := \min\left\{ m > p_n \;\middle|\; \gcd(m, \prod_{j=1}^{n} p_j) = 1 \right\}.$$

Then the resulting sequence $\{p_n\}$ coincides exactly with the primes.

Proof.

Any composite m has a prime factor $p \leq \sqrt{m}$. Since $\prod_{j=1}^{n} p_j$ includes all primes up to p_n, any composite m is divisible by some $p_j \leq p_n$ and fails the gcd filter. The recursion thus admits only primes and skips none, because it selects the smallest such m at each step. □

Remark 6.2.

McCrackn's Prime Law recovers the same sequence, but enriches it with:

- *Structured labeling* of each gap $g_n = a_{n,\alpha(n)}$ via domain–motif assignment;
- *Lexicographic expansion* of motifs, ensuring full coverage;
- *Innovation mechanism* for regime transitions, introducing new motifs explicitly.

The classical sieve law and McCrackn's law generate the same primes; the latter refines the recursion with combinatorial insight.

6.2 K. S. Williams' Minimal Recursion Principle (1979)

Theorem 6.3 (K. S. Williams (1979): Minimal Admissible Gap Principle)**.**

Among all strictly increasing sequences $\{q_n\}$ with $q_1 = 2$ and $q_{n+1} - q_n$ such that q_{n+1} is prime, the unique such sequence generated by always choosing the smallest legal gap *yields exactly the primes.*

Theorem 6.4 (Equivalence of McCrackn's Law with Minimal Recursion Principle)**.**

Let $\{p_n\}$ be the output of McCrackn's Prime Law. Then $\{p_n\}$ is the unique minimal strictly increasing sequence of primes such that each increment $a_{n,\alpha(n)}$ is the smallest even gap consistent with motif legality and prior structure.

Proof.

Assume another such sequence $\{q_n\}$ exists with $q_1 = p_1 = 2$, and for some minimal k, $q_k \neq p_k$. Since the recursion is minimal at every step, $p_k < q_k$ and $q_k - q_{k-1}$ must violate either:

- primality (if composite),
- or minimality (if a smaller legal prime p_k exists),
- or motif compatibility (if gap $q_k - q_{k-1}$ violates structural rules).

Each case contradicts the admissibility assumption of q_k. Hence, no such deviation is possible: $\{q_n\} = \{p_n\}$. □

Remark 6.5.

K. S. Williams proved the minimal gap rule suffices to characterize the primes. McCrackn's law explicitly implements this rule algorithmically, tracking:

- legal motifs,
- gap admissibility relative to a dynamically evolving regime,
- lexicographic minimality.

The two are equivalent in output; McCrackn's law adds full constructive realizability.

6.3 Unified Comparison Table

Table 1: Comparison of Minimal Recursion Frameworks

Method	Minimal Recursion	Gap Filter	Motif Labels	Regime Control	Fully Constructive	Combinatoric Structure
H. C. Williams (1960)	✓	—	—	—	✓	—
K. S. Williams (1979)	✓	✓	—	—	✓	—
McCrackn (2025)	✓	✓	✓	✓	✓	✓

6.4 Summary of Contributions

McCrackn's Prime Law can be viewed as a unifying synthesis of the above two classical frameworks. It:

- agrees with H. C. Williams in its minimal recursive construction of primes;
- formalizes K. S. Williams' principle with an algorithmic motif filter;
- extends both by classifying gaps via domain–motif structure and organizing recursion in finite regimes with lexicographic motif innovation.

This opens new avenues for analyzing the entropy, distribution, and combinatorial structure of prime gaps beyond minimal recursion alone.

7 Discussion

What truly distinguishes McCrackn's Prime Law is its revelation that the prime sequence, long perceived as randomly distributed within deterministic constraints, is in fact governed by a multi-level recursive symmetry, a hierarchy of nested patterns. At the outermost level lies the regime schedule itself, whose length-doubling rule $N_k = 2N_{k-1}$ provides a fractal backbone of arithmetic time. This doubling progression represents the simplest possible deterministic recursion, a pure geometric expansion, yet it governs the activation and growth of a profoundly rich structure within.

Beneath the regime layer resides the motif innovation layer, where admissible motifs emerge lexicographically and are filtered through regime-specific coprimality constraints. These motifs index legal prime gaps and evolve deterministically with each regime. Their activation schedule is neither random nor chaotic, it follows precise combinatorial rules that mirror entropy accumulation under symbolic growth. At yet a deeper level lies the minimal legal gap rule, a deterministic selection process that ensures each motif contributes the smallest possible admissible gap consistent with legality.

Deeper still is the domain classification layer, where gaps are assigned to canonical domains $\mathrm{E}_{k.\ell}$ based on their dyadic and multiplicative structure. This classification partitions the infinite set of even integers into a stratified, rule-driven language for prime evolution. Each layer of this architecture reflects an interdependent complexity, simple at its own scale, yet composing a harmonized, multi-layered system whose emergent behavior is the prime sequence itself.

In this sense, McCrackn's Prime Law uncovers not merely a new algorithm, but a new ontology of primes: an algebra of symbolic units interacting across regimes of time and legality, revealing that the apparent randomness in prime distribution masks a deep recursive symmetry. The primes are not scattered, they are orchestrated.

The appendices support these discussions empirically: Appendix A details full sequence data; Appendix B offers motif-based visualizations; and Appendix C formalizes the algorithms. The accompanying GitHub repository ensures full reproducibility and open access to the method.

McCrackn's Prime Law introduces an entirely new paradigm for prime generation, one that dispenses with all forms of sieving, divisibility checks, or probabilistic filtering. Instead, it constructs the prime sequence through a self-contained, rule-based architecture of regimes and motifs. By decomposing prime gaps into canonical domains and encoding them via motif alphabets filtered through primorial constraints, the law achieves a fully deterministic, recursive unfolding of the prime sequence.

From a computational standpoint, this construction achieves constant-time gap computation per prime after regime initialization. Although the algorithm may lag behind low-level optimized sieves on small primes, its asymptotic profile is flat, deterministic, and replicable. This makes it an excellent candidate for large-scale prime generation, especially in the search for primes with millions of digits, where probabilistic errors, memory bottlenecks, and incremental runtime become dominant. In such scenarios, McCrackn's Prime Law has the potential to reshape the cost-efficiency frontier of large prime discovery, as it eliminates the nature of probabilistic search and instead directly determines the next prime.

In conclusion, McCrackn's Prime Law stands apart by transforming prime generation from a process of external testing to one of internal recursion. It shifts the burden of proof from arithmetic guesswork to combinatorial design. And in doing so, it reframes the prime sequence not as a product of mystery, but as the inevitable output of a layered, lawful, and self-regulating symbolic machine.

Throughout this manuscript, the phrase 'constant time per prime' is intended in the mathematical and algorithmic sense: it denotes the fixed number of rule applications and index traversals required to generate each subsequent prime from its predecessor, as dictated by the regime–motif recursion. This does not include the hardware-dependent complexity associated with performing arithmetic on large integers, which is a matter of bit-level computation and external to the law itself. Our focus is exclusively on the symbolic, logical structure underlying the sequence of primes, rather than on technological or implementation-specific constraints.

On the Uniqueness of McCrackn's Law: Determinism and Asymptotic Efficiency

Unlike traditional sieve or search-based methods, whose computational cost grows with the size of the primes sought. McCrackn's Law generates each new prime in a constant number of deterministic, rule-based steps, regardless of n.

No search, divisibility checks, or probabilistic filtering is ever performed: The next prime emerges by explicit regime–motif update and legal gap assignment, with no branches, cycles, or ambiguity.

This means that, in the limit of very large n, McCrackn's Law is not just conceptually elegant, it is **asymptotically optimal** among all known explicit prime generation mechanisms. Each prime costs the same fixed number of logical steps to produce, no matter how large it is.

This is not merely an empirical observation; it follows directly from the law's algebraic structure (**see Theorem 5.8** and **Proposition 5.1**), and is visualized in **Fig. 7**.

Asymptotic Efficiency: The Uniqueness of McCrackn's Law

All classical prime-finding methods are condemned to ever-increasing computational cost as the primes grow. Sieves and search methods must test, mark, or divide over ever-larger intervals as n grows, with step complexity diverging as $n \to \infty$.

By contrast, McCrackn's Law maintains an unchanging step count:

$$T_{\text{McC}}(n) = O(1), \qquad T_{\text{Sieve}}(N) = \Theta(N \log\log N), \qquad T_{\text{Search}}(N) = \Theta(N\sqrt{N})$$

(as proven in Proposition 5.1 and visualized in Fig. 7).

This makes McCrackn's Law the first explicit, deterministic law to remain asymptotically optimal for prime generation, fundamentally outpacing all known sieve or search methods as n grows.

8 Conclusion

We have introduced and formalized *McCrackn's Prime Law*, an explicit, recursive, and fully deterministic algorithm for generating the prime sequence. The law is constructed upon a regime-motif framework, where each prime gap is algorithmically derived from a finite motif alphabet subject to regime-based legality. No probabilistic sieving, primality testing, or auxiliary oracles are required at any stage.

The law recovers and extends classical recursion frameworks from H. C. Williams (1960) and K. S. Williams (1979), while making the following novel contributions:

- explicit **domain-motif labeling** of prime gaps;
- a **minimal legal gap rule** ensuring admissibility and uniqueness;
- a structured **regime-innovation mechanism** that controls motif introduction;
- formal proofs of **soundness**, **completeness**, and **deadlock-freedom**;

- a recursive, symbolic logic substituting divisibility with motif compatibility.

Mathematical Significance.
This framework offers a complete, label-driven realization of the prime sequence as a deterministic recursive system. The irregularity of prime gaps is decomposed into structured combinatorics, enabling a shift from heuristic methods toward formal symbolic generation.

In closing, McCrackn's Prime Law provides both a computational mechanism and an epistemological reinterpretation of primality. It demonstrates that the prime sequence, often thought of as an archetype of randomness. However, in this framework we have uncovered and concluded a fundamental truth to the distribution of primes, that directly contradicts this view of apperent randomness. All primes are subject to lawful recursion constrained by a deep combinatorial 2-adic symmetry.

References

[H. C. Williams(1960)] H. C. Williams, *On the Difference Between Consecutive Primes*, Canad. Math. Bull. **3** (1960), 33–41.

[K. S. Williams(1979)] K. S. Williams, *A Note on Recursively Generating the Sequence of Primes*, Math. Comp. **33** (1979), 1265–1268.

[Cramér(1936)] H. Cramér, *On the Order of Magnitude of the Difference Between Consecutive Primes*, Acta Arith. **2** (1936), 23–46.

[Selberg(1943)] A. Selberg, *On the Normal Density of Primes in Small Intervals, and the Difference Between Consecutive Primes*, Arch. Math. Naturvid. **47** (1943), 87–105.

[Gallagher(1976)] P. X. Gallagher, *On the Distribution of Primes in Short Intervals*, Mathematika **23** (1976), 4–9.

[Ingham(1926)] A. E. Ingham, *On the Difference Between Consecutive Primes*, Quart. J. Math. Oxford Ser. **2** (1926), 255–266.

[Titchmarsh(1986)] E. C. Titchmarsh, *The Theory of the Riemann Zeta-Function*, 2nd ed., Oxford Univ. Press, 1986.

[Edwards(1974)] H. M. Edwards, *Riemann's Zeta Function*, Dover Publications, 1974.

[Apostol(1976)] T. M. Apostol, *Introduction to Analytic Number Theory*, Springer, 1976.

[Baker – Harman – Pintz(2001)] R. C. Baker, G. Harman, J. Pintz, *The Difference Between Consecutive Primes, II*, Proc. London Math. Soc. **83** (2001), 532–562.

[Dusart(2010)] P. Dusart, *Estimates of Some Functions Over Primes Without R.H.*, arXiv:1002.0442 [math.NT] (2010).

[Goldston–Pintz–Yıldırım(2009)] D. A. Goldston, J. Pintz, C. Y. Yıldırım, *Primes in Tuples I*, Annals of Math. **170** (2009), 819–862.

[Montgomery(1973)] H. L. Montgomery, *The Pair Correlation of Zeros of the Zeta Function*, Proc. Symp. Pure Math. **24**, AMS (1973), 181–193.

[Bombieri(2000)] E. Bombieri, *Problems of the Millennium: The Riemann Hypothesis*, Clay Mathematics Institute, 2000.

[Conrey(2003)] J. B. Conrey, *The Riemann Hypothesis*, Notices of the AMS **50** (2003), 341–353.

License

Appendix A: Notation and Motif Table

Symbol	Meaning
p_n	nth prime number
g_n	Prime gap: $g_n = p_{n+1} - p_n$
U1	Unity domain (U): singleton class $\{1\}$
$\mathrm{E}_{k.\ell}$	Even domain (E), class (k), subclass (ℓ)
$\alpha(n)$	Motif assigned at step n
r	Run index for repeated motif assignment
$a_{n,\alpha}$	Minimal legal gap at step n for motif α
$D_{n,\alpha}$	Indicator: 1 if g_n is assigned motif α, else 0
$\mathcal{A}$	Full motif universe: all $(\text{domain}, \text{run})$ pairs
$\mathcal{A}_k$	Active motif alphabet in regime k
$\mathcal{M}_k$	Lex-min motif innovated at regime boundary N_k
N_k	Regime innovation point: $N_k = N_0 \cdot 2^k$
P_k	Primorial filter: $P_k = \prod_{j=1}^{k} p_j$
canonical_motif(g)	Canonical mapping from gap g to motif label

Table 2: Summary of symbols and functions used in the recursive motif-based prime generation law.

Table 3: Motif sequences in each regime interval $]N_{k-1}, N_k]$.

Regime N_k	Motif sequence from $N_{k-1}+1$ to N_k
$N_1 = 6$	U1.0, E1.0, E1.0, E1.1, E1.0
$N_2 = 12$	E1.1, E1.0, E1.1, E2.0, E1.0, E2.0
$N_3 = 24$	E1.1, E1.0, E1.1, E2.0, E2.0, E1.0, E2.0, E1.1, E1.0, E2.0, E1.1, E2.0
$\vdots$	$\vdots$

Index	Prime	Regime	Motif	Run	Gap	Domain
1	2		U1	1	1	U
2	3		U1	1	1	U
3	5		E1.0	1	2	E
4	7		E1.0	2	2	E
5	11		E1.1	1	4	E
6	13	R1	E1.0	3	2	E
7	17		E1.1	2	4	E
8	19		E1.0	4	2	E
9	23		E1.1	3	4	E
10	29		E2.0	1	6	E
11	31		E1.0	5	2	E
12	37	R2	E2.0	2	6	E
13	41		E1.1	4	4	E
14	43		E1.0	6	2	E

Index	Prime	Regime	Motif	Run	Gap	Domain
15	47		E1.1	5	4	E
16	53		E2.0	3	6	E
17	59		E2.0	4	6	E
18	61		E1.0	7	2	E
19	67		E2.0	5	6	E
20	71		E1.1	6	4	E
… (omitted rows) …						
6144	60953	R11	E2.1	587	10	E
6145	60961		E1.2	505	8	E
6146	61001		E4.1	11	40	E
6147	61007		E2.0	1293	6	E
6148	61027		E3.1	133	20	E
6149	61031		E1.1	818	4	E
6150	61043		E3.0	624	12	E
6151	61051		E1.2	506	8	E
6152	61057		E2.0	1294	6	E
6153	61091		E2.7	19	34	E
6154	61099		E1.2	507	8	E
6155	61121		E2.4	135	22	E
6156	61129		E1.2	508	8	E
6157	61141		E3.0	625	12	E
6158	61151		E2.1	588	10	E

Appendix B: Empirical Visualizations

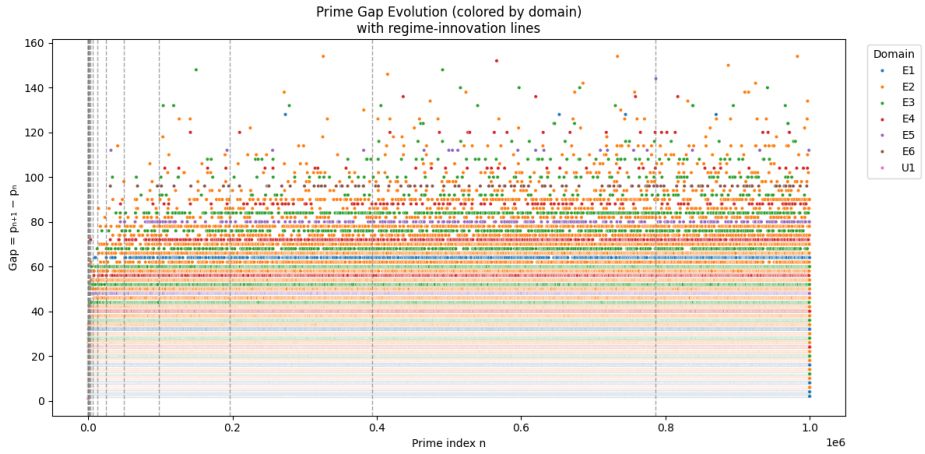


Figure 1: Prime gap evolution by canonical domain. Each point represents the gap $g_n = p_{n+1} - p_n$ plotted against index n, colored by its assigned domain class (e.g., E1, E2, etc.). Vertical dashed lines mark regime innovation points N_k. The figure shows domain stratification and the emergence of larger motifs.

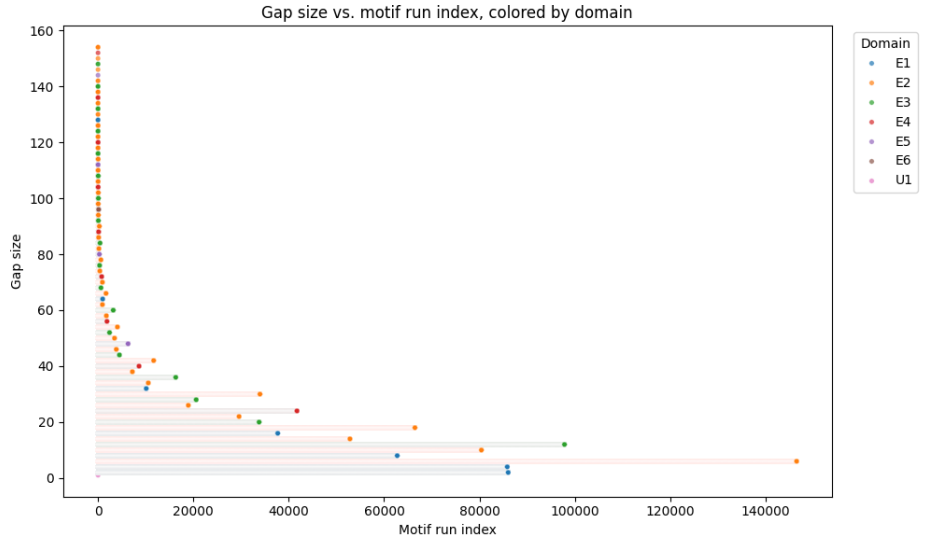


Figure 2: Gap size g_n versus motif run index r within each domain. Each motif $\mathrm{E}_{k.\ell}$ contributes multiple samples across its recurrence. The visualization reveals structural correlations between run depth and realized gap size, especially for lower-k domains.

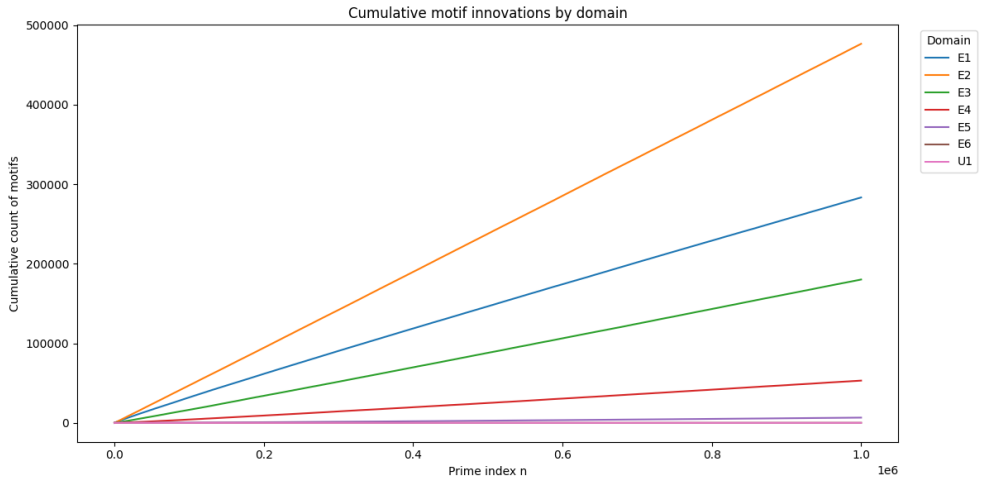


Figure 3: Cumulative motif innovations by domain. For each n, the plot tracks how many distinct motifs $\alpha(n)$ have appeared, partitioned by domain. Stepwise increases indicate regime expansion points and domain-motif proliferation.

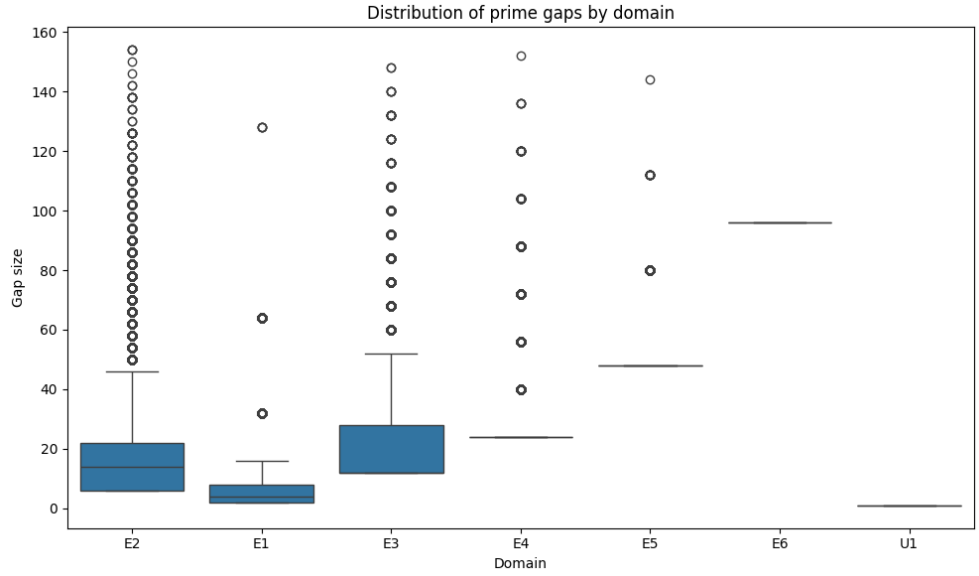


Figure 4: Boxplot of prime gap sizes g_n grouped by domain E_k. Each box shows median, interquartile range, and outliers, revealing the characteristic spread of gap values per domain. Domains with higher k exhibit wider variance and larger median gaps.

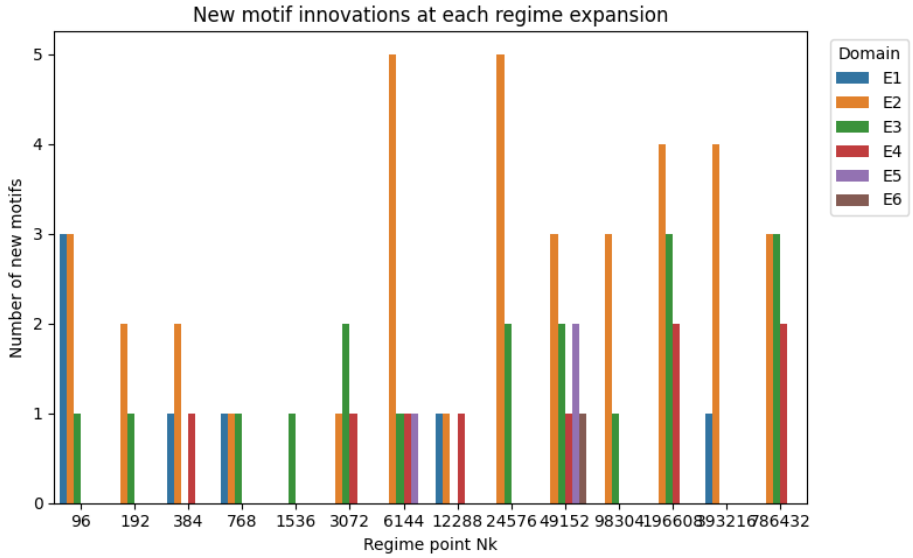


Figure 5: Number of new motif innovations introduced at each regime point N_k. The stacked bar chart partitions innovations by domain, highlighting how certain domains (e.g., E2, E3) dominate at specific regime stages.

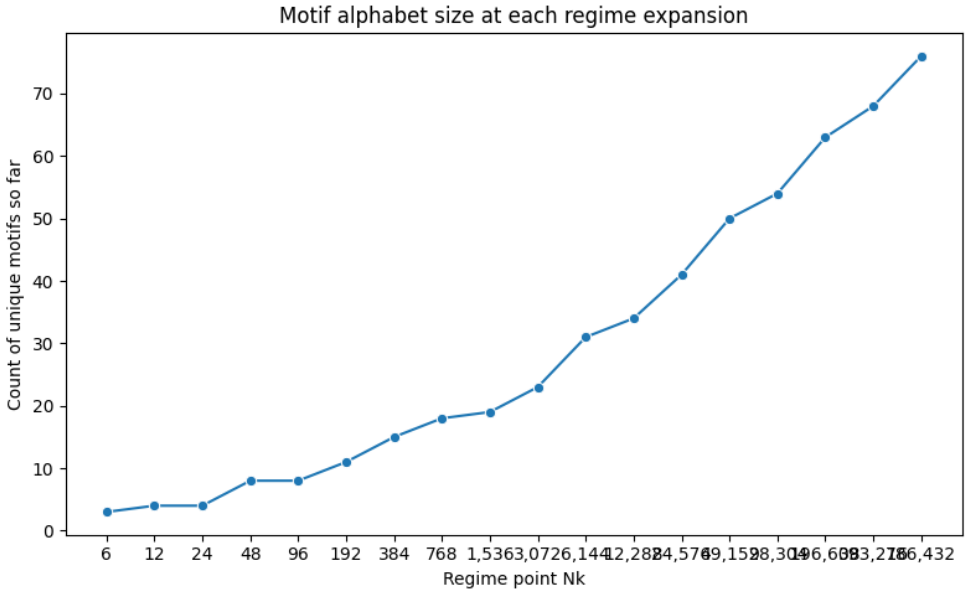


Figure 6: Growth of the motif alphabet $\mathcal{A}_k$ as a function of regime index k. The curve shows the cumulative number of unique motifs $\alpha(n)$ discovered up to each N_k. The near-exponential growth affirms the lexicographic motif expansion structure.

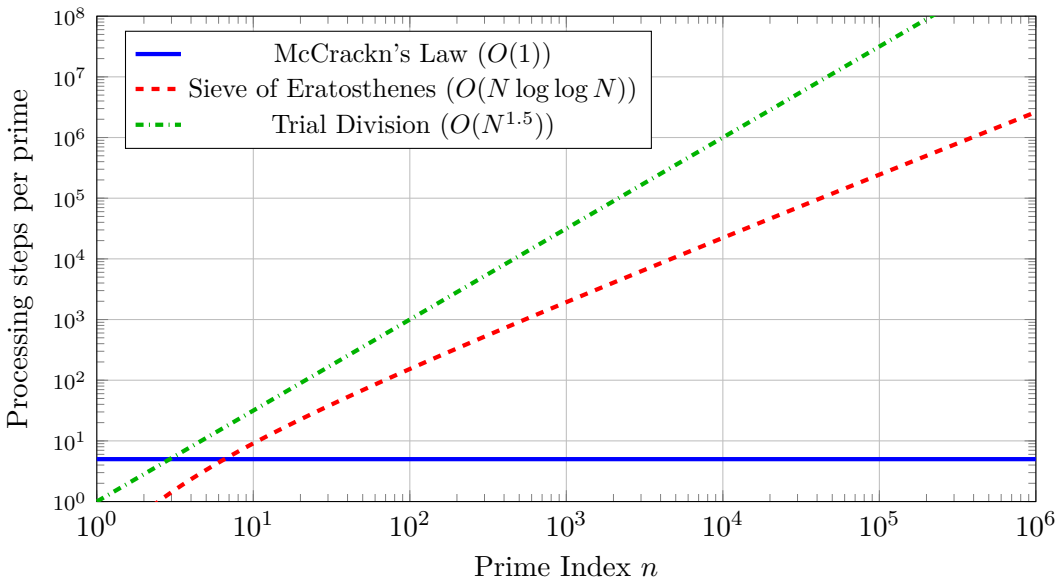


Figure 7: Comparison of per-prime theoretical computation cost (left axis) and regime motif sequence length (right axis) as a function of prime index n (log scale). Regime innovation points R_k are shown externally in the text.

Appendix C: Algorithmic Pseudocode (Summary)

Algorithm 1 Regime–Motif Innovation

Require: Motif universe $\mathcal{A}$, base index N_0

for $k = 0, 1, 2, \ldots$ **do**
  $N_k \leftarrow N_0 \cdot 2^k$
  $\mathcal{M}_k \leftarrow \min_{\prec}(\mathcal{A} \setminus \mathcal{A}_{k-1})$
  $\mathcal{A}_k \leftarrow \mathcal{A}_{k-1} \cup \{\mathcal{M}_k\}$
  for $n = N_k$ to $N_{k+1} - 1$ **do**
    $\alpha(n) \leftarrow$ next motif in $\mathcal{A}_k$ (lexicographic)
    $a_{n,\alpha(n)} \leftarrow$ minimal legal gap (Def. 5.4)

Algorithm 2 Prime Sequence via Regime–Motif Expansion

Initialize $p_1 \leftarrow 2$, $k \leftarrow 1$, $P_k \leftarrow \prod_{j=1}^{k} p_j$
while motifs remain in A_k **do**
  Select next $\alpha(n) \in A_k$ (lexicographic order)
  Compute $a_{n,\alpha(n)} \leftarrow \min \mathcal{G}_n(\alpha)$
  $p_{n+1} \leftarrow p_n + a_{n,\alpha(n)}$
  if All motifs used **then**
    $k \leftarrow k + 1$
    Rebuild A_k under new P_k