Computation and Cryptographic Fingerprinting of a 109-Million-Digit Integer in the Q_{59} Polynomial Family

Ruqing Chen GUT Geoservice Inc., Montreal, Canada ruqing@hotmail.com January 3, 2026

📄 **PUBLICATION & CODE LINKS**

Permanent DOI:

https://doi.org/10.5281/zenodo.18140813

Code & Data Repository:

https://github.com/Ruqing1963/q59-prime-candidate

Abstract

This paper documents the successful computation of an extremely large integer generated by the polynomial $Q_n(q) = q^n - (q-1)^n$ at specific parameters n=59 and $q=10^{1886792} + 3$. The resulting value contains precisely 109,433,938 decimal digits. We provide the SHA-256 cryptographic fingerprint to establish unique identity and enable independent verification. **All computational scripts, verification tools, and complete documentation are publicly available on GitHub at https://github.com/Ruqing1963/q59-prime-candidate.** This work represents a computational achievement in handling ultra-large integers and serves as a foundation for future primality verification efforts. **We emphasize that primality testing has not yet been performed on this number; it remains a candidate requiring rigorous verification through distributed computational resources.**

1. Introduction

The polynomial family $Q_n(q) = q^n - (q-1)^n$ has been studied extensively in number theory due to its relationship with prime number generation and the Bunyakovsky conjecture [1]. Through binomial expansion, this polynomial can be expressed as:

$$Q_n(q) = nq^{n-1} - C(n,2)q^{n-2} + ... + (-1)^{n-1}$$

This structural property makes the polynomial family particularly interesting for computational number theory research. Our previous work [2] documented systematic searches across this family, discovering over 4,000 probable primes with the largest being 5,987 digits at n=47.

The present work extends this research by computing a dramatically larger candidate at n=59 with a base q of unprecedented magnitude. While this represents a significant computational achievement, we acknowledge that formal primality verification remains an open challenge requiring substantial computational resources beyond those currently available to us.

To ensure full transparency and enable independent verification, all computational scripts, verification tools, and complete documentation are publicly available in an open-source repository:

https://github.com/Ruqing1963/q59-prime-candidate

2. Mathematical Construction

2.1 Parameter Selection

The specific candidate was constructed using the following parameters:

- **Exponent:** $n = 59$ (a prime exponent)
- **Base:** $q = 10^{1886792} + 3$
- **Offset:** 3 (selected to avoid certain divisibility patterns)

The complete formula for the candidate number N is:

$$\mathbf{N = (10^{1886792} + 3)^{59} - (10^{1886792} + 2)^{59}}$$

2.2 Expected Digit Count

For large q, the leading term of $Q_n(q)$ is nq^{n-1}. The expected digit count can be approximated as:

$$\text{Digits} \approx (n-1) \times \log_{10}(q) = 58 \times 1{,}886{,}792 = 109{,}433{,}936$$

Our computed value has 109,433,938 digits, differing by only 2 digits (0.000002%) from the theoretical estimate, confirming the computational accuracy.

3. Computational Methodology

3.1 Parameter Selection Strategy

The selection of parameters ($n=59$, $q=10^{1886792} + 3$) was not arbitrary. Several strategic considerations guided this choice:

- **Prime exponent:** The value n=59 was chosen as a prime number to avoid systematic divisibility patterns that affect composite exponents.
- **Base magnitude:** The base q was selected to produce a result exceeding 100 million digits, aligning with the EFF prize threshold while remaining computationally feasible to calculate.
- **Offset selection:** The offset of +3 was chosen based on preliminary analysis to avoid certain small prime divisors that would immediately disqualify the candidate.

Our previous research [2] on the Q_n family informed these choices, though we emphasize that parameter selection does not constitute primality verification. Even carefully chosen candidates must undergo rigorous testing.*

**The heuristic motivation for targeting this specific candidate was based on preliminary analysis of patterns in the Q_n family's behavior, which may relate to zero-distribution patterns in associated L-functions. A rigorous theoretical framework for this approach will be detailed in future work.*

3.2 Hardware and Implementation Environment

The computation was performed on a standard x86-64 workstation with 32GB RAM and an Intel Core processor, demonstrating that Python's arbitrary-precision integer arithmetic can handle ultra-large computations without requiring specialized supercomputing infrastructure for the calculation phase (though such infrastructure would be essential for primality verification).

3.3 Python Implementation

The computation was implemented in Python 3.11, utilizing its arbitrary-precision integer arithmetic capabilities. A critical technical requirement was removing Python's default string conversion limit for large integers:

```
import sys
sys.set_int_max_str_digits(0)
```

This configuration enabled the conversion of the 109-million-digit integer to its complete decimal string representation, which was essential for both verification and fingerprinting.

3.4 Computation Process

The computation proceeded through the following stages:

1. **Base construction:** Calculate $q = 10^{1886792} + 3$ using Python's native power operator
2. **Polynomial evaluation:** Compute q^{59} and $(q-1)^{59}$
3. **Final calculation:** Subtract to obtain $N = q^{59} - (q-1)^{59}$
4. **String conversion:** Convert the integer to decimal string format (975 seconds)
5. **Fingerprinting:** Calculate SHA-256 hash of the complete decimal representation

The total processing time for computation and fingerprinting was approximately 975 seconds (16.25 minutes) on standard hardware.

4. Computational Results and Identity Certificate

The computed integer and its cryptographic fingerprint are documented in Table 1. This fingerprint serves as a unique mathematical identity that can be independently verified by recomputing the same polynomial value.

Property	Value
Formula	$Q_{59}(10^{1886792} + 3)$
Decimal Digits	109,433,938
SHA-256 Hash	`A462032F37DC907EDEF64A575696EC0B11C052B42B469C76F6A6577B841864EF`
First 60 Digits	`590000000000000000000000000000000000000000000000000000000000`
Last 60 Digits	`000000000000000000000000000000014130386091162273752461387579`
Computation Date	January 3, 2026

Table 1: Computational Results and Identity Certificate

5. Primality Verification Status

5.1 Current Status

We must state clearly and unambiguously: ***No primality testing has been performed on this number.*** The value documented in this paper is a *computational candidate* that requires rigorous verification through established primality testing protocols.

The number's membership in the Q_n polynomial family provides mathematical interest, but does not constitute evidence of primality. While our previous research [2] demonstrated that this family can generate probable primes, each candidate must be individually verified.

5.2 Computational Challenges

For a number of 109,433,938 decimal digits (approximately 363 million bits), even probabilistic primality testing presents extraordinary computational challenges:

- **Miller-Rabin Test:** A single round requires modular exponentiation of a number this size. Standard practice requires 20+ independent rounds for high confidence, each taking days to weeks on specialized hardware.
- **Lucas Pseudoprime Test:** The Lucas component of the Baillie-PSW test involves comparable computational complexity.
- **Deterministic Proof:** ECPP (Elliptic Curve Primality Proving) for a number this size is beyond current computational feasibility. No number of this magnitude has been proven prime using deterministic methods.

For context, the largest known proven primes are Mersenne primes (of the form 2^p-1), which benefit from specialized testing algorithms. The Great Internet Mersenne Prime Search (GIMPS) requires weeks to months per candidate even with these optimizations. General numbers like ours face significantly greater challenges.

5.3 EFF Prize Considerations

The Electronic Frontier Foundation offers prizes for the discovery of large primes, including a $150,000 award for the first *proven* prime with at least 100 million digits [3]. While our candidate exceeds this digit threshold, it **does not currently qualify** for consideration because:

- The EFF requires *proven primes*, not probable primes or untested candidates
- Independent verification is required
- Complete documentation of primality proofs must be provided

We present this candidate as a *potential* path toward such a discovery, contingent on successful verification that has not yet been attempted.

6. Future Work and Call for Collaboration

The verification of this candidate requires computational resources substantially beyond those available to individual researchers. We invite collaboration from institutions with access to:

- **Supercomputer facilities:** Distributed computing clusters capable of sustained multi-week calculations
- **Specialized algorithms:** Expertise in optimized implementations of Miller-Rabin and Lucas tests for ultra-large integers
- **Independent verification:** Multiple parties independently testing the same candidate using the SHA-256 fingerprint for identification

The SHA-256 hash provided in this paper uniquely identifies this specific number. Any researcher can independently compute $Q_{59}(10^{1886792} + 3)$ and verify the fingerprint matches, establishing that they are testing the same candidate.

Additionally, our methodology for computing values in the Q_n family at this scale may prove useful for exploring other candidates that could be more computationally tractable for primality verification.

7. Conclusion

We have successfully computed and fingerprinted a 109-million-digit integer from the Q_{59} polynomial family. This work demonstrates the feasibility of handling numbers of this magnitude using widely available programming tools and documents a specific candidate for future primality research.

If successfully verified as prime, this candidate would represent a potential qualification for the Electronic Frontier Foundation's $150,000 Cooperative Computing Award for the first proven 100-million-digit prime. However, such verification would require extraordinary computational resources and represents a substantial challenge for the mathematical community.

However, we emphasize that **computation is not verification**. This number's primality status remains completely unknown and will require substantial collaborative effort to determine. We present this work as a contribution to the computational exploration of the Q_n family and as an invitation for collaborative verification efforts.

The cryptographic fingerprint and sample digits provided ensure that this specific candidate is uniquely identified and can be independently reproduced, establishing a foundation for any future verification work that may be attempted.

8. Code and Data Availability

Complete Source Code Repository:

https://github.com/Ruqing1963/q59-prime-candidate

The GitHub repository contains:

- `compute_q59.py` - Complete computation script with built-in verification
- `verify_fingerprint.py` - SHA-256 verification tool
- Detailed methodology documentation
- Verification guides and collaboration guidelines
- Identity certificate and sample outputs

All code is released under the Creative Commons Attribution 4.0 International (CC-BY-4.0) license, enabling free use with appropriate attribution. The repository is actively maintained and welcomes community contributions through GitHub's issue tracking and pull request systems.

For direct inquiries or collaboration proposals: ruqing@hotmail.com

Acknowledgments

The author thanks the Python Software Foundation for maintaining the arbitrary-precision integer arithmetic capabilities that made this computation possible.

References

[1] Bunyakovsky, V. (1857). "Sur les diviseurs numériques invariables des fonctions rationnelles entières." *Mém. Acad. Sci. St. Pétersbourg*.

[2] Chen, R. (2025). "Computational Verification of the Bunyakovsky Conjecture for the Polynomial $Q_n(q) = q^n - (q-1)^n$." *Unpublished manuscript*, December 21, 2025.

[3] Electronic Frontier Foundation. "Cooperative Computing Awards." https://www.eff.org/awards/coop. Accessed January 3, 2026.