A Mathematical Reformulation of the Collatz Conjecture: Final Report and Roadmap

Author: Imad Aissani — Université de Guelma

Contact: imadarc95@gmail.com

Date: 2025-10-09 UTC

Executive Summary

We developed a hybrid analytic–computational framework that reformulates Collatz trajectories via the Diophantine identity $2^S C = 3^r N + 1$ and its logarithmic form $\Lambda = r\cdot\ln 3 - S\cdot\ln 2 = \ln(1 + 1/(3^r N))$. This document summarizes work completed, derivations, computational workflow, and the advanced experimental stage where the project paused pending large-scale runs.

1. Key derivation (step-by-step)

Start from an initial integer N and consider a finite prefix of its Collatz trajectory that applies r odd steps (3n+1) and interleaved halving runs. Let $b_j \geq 0$ be the number of immediate halving steps after the j-th odd step. Define $B_j = \sum_{i=1}^j b_i$ and $S = \sum_{i=1}^r b_i$ the total number of halvings. Writing the contribution of each odd step in place value yields: $3^r N + \sum_{j=1}^r 3^{r-j} 2^{B_j} = 2^S C$, where $C = \sum_{j=1}^r 2^{B_j - S}$ is a positive integer aggregating the shift-pattern. Rearranging gives $2^S C = 3^r N + 1$.

2. Logarithmic linear form and Matveev application

From the main identity divide both sides by $3^r N$ and take logarithms to obtain $\Lambda = r \ln 3 - S \ln 2 = \ln(1 + 1/(3^r N))$. Matveev's theorem gives explicit lower bounds for $|\Lambda|$ (if nonzero) in terms of heights. The computational strategy compares an upper bound $U = \ln(1 + 1/(3^r N))$ with a Matveev lower bound L_{mat}; if $L_{mat} > U$ the pair is ruled out.

3. Detailed computational workflow (step-by-step)

1. Theoretical preprocessing: compute convergents of $\ln 3/\ln 2$ and estimate thresholds (R_0,S_0) via Matveev inequalities.
2. Modular sieving: for primes p up to P_{max} precompute residues $2^S \bmod p$ and $3^r \bmod p$; eliminate mismatches.
3. Baker–Davenport reduction: use continued fraction convergents to shrink windows.
4. Beam-guided DP search: construct C via DP, prune by residue scores.
5. ECM factorization: factor $D = 2^S - 3^r$ where helpful.
6. LLL refinement: use lattice reduction to find contradictions.
7. Distributed execution: split r ranges across nodes with checkpointing.
8. Final certification: emit witnesses or exclusion certificates.

4. Flowchart (textual)

Preprocess → Modular Sieve → Baker–Davenport → Beam DP → ECM / LLL → Distributed Run → Certification

5. Implementation deliverables and commands

We prepared a C/GMP OpenMP implementation with checkpointing (file: collatz_beam.c). Recommended build and run commands:

```
$ gcc -O3 -march=native -fopenmp -lgmp -o collatz_beam collatz_beam.c
$ ./collatz_beam -r 31867 -S 50508 -block 50 -beam 200 -threads 16 -checkpoint
/data/ckpt.tsv -runtime 43200
```

Checkpoint files are plain-text and contain hex-coded mpz numbers for independent replay.

6. Proof sketches and rigorous components

A. Exact derivation of main identity is algebraic and contains no approximation.
B. Upper bound: $|\Lambda| \le 1/(3^r N)$.
C. Matveev lower bound: existence of effective lower bound Q^{-A} for nonzero Λ. Compute Q^{-A} and compare with upper bound.
D. If Matveev lower bound exceeds upper bound, contradiction, hence parameter excluded.

7. Resources, timeline and checkpoints

Minimal recommended cluster: 16 cores, 64 GB RAM, 1 TB SSD. Suggested timeline: preparatory analysis 1 week; modular sieving 2–7 days; beam-search 3–10 days; final LLL/Matveev tightening 1–2 weeks.

Contact and next steps

Author: Imad Aissani — Université de Guelma
Contact email: imadarc95@gmail.com

Next step: provision an HPC instance or provide system access to run distributed checkpoints; alternatively we package the executable and scripts for immediate deployment.

---- End of report ----