

Algorithmic Quantification of Balanced Ternary Arithmetic: A Software-Based Comparative Study

Nauman Sajjad, Ahmad Sharjeel, Mahwish Abid
Dept. of Computer Science, Lahore Garrison University

Abstract

Binary arithmetic forms the computational foundation of modern digital systems, providing robustness through simplicity and transistor compatibility. However, as the physical limits of binary scaling constrain hardware progress, alternative logic representations have gained relevance. This study explores balanced ternary arithmetic, an elegant, symmetric logic system utilizing three states ($-1, 0, +1$), as a potentially superior model for low-complexity computation.

By analyzing seminal studies on ternary architectures and implementing a comparative, software-based quantification model, this paper investigates the operational efficiency, arithmetic simplicity, and logic reduction benefits inherent in balanced ternary design. Results from Python-based simulations validate that ternary arithmetic reduces carry propagation, circuit depth, and logical redundancy compared to binary systems. Empirical findings show up to 36% fewer primitive operations and 37% faster execution for addition, confirming theoretical efficiency.

Keywords

Balanced ternary logic, multi-valued computation, arithmetic simulation, CNTFET, memristors, computational complexity, AI efficiency, quantum-inspired logic.

I. Introduction

Binary logic, characterized by two discrete voltage levels representing 0 and 1, has dominated digital computing since the advent of transistor-based systems [1]. While simple and reliable, binary arithmetic now faces scalability bottlenecks as transistor miniaturization approaches physical limits. The balanced ternary system, using three

symmetric states ($-1, 0, +1$), encodes more information per digit and eliminates sign bits, naturally representing positive and negative values [2][3].

Although early ternary computers such as Setun (1958, USSR) demonstrated theoretical promise, hardware technology of the time limited adoption. Modern devices like CNTFETs and memristors have revived interest in multi-valued logic [4][5]. Yet, there remains a lack of software-level quantification of ternary arithmetic performance, independent of circuit fabrication.

This study addresses that gap by simulating binary and balanced ternary arithmetic addition, subtraction, and multiplication, through a Python-based comparative model, quantifying operation counts, carry propagation, and execution times.

II. Literature Review

Ahmad and Alam [1] introduced mathematical models for balanced ternary arithmetic, showing simplified signed computation. Wang et al. [2] demonstrated that CNTFET-based ternary gates reduce transistor count by up to 30%. Zahoora et al. [3] extended this analysis, reporting up to 25% lower energy-delay product in ternary architectures.

Further studies linked ternary arithmetic with quantum qutrit systems and AI accelerators [6][7], suggesting that arithmetic symmetry reduces operation depth and energy cost. However, these analyses remained hardware centric.

To bridge theory and implementation, this paper develops a software-based quantification framework, extending Sharma and Singh's [8] algorithmic analysis into ternary arithmetic.

III. Comparative Methodology

A. Conceptual Framework

Arithmetic operations, addition, subtraction, and multiplication, are simulated in both binary and balanced ternary logic. The following parameters are measured:

- N_ops: total primitive operations
- D_carry: maximum carry propagation depth
- T_exec: mean execution time per operation

The conceptual symmetry of ternary digits ($-1, 0, +1$) is illustrated in Table 1, showing direct representation of negative values without separate sign bits.

Dec	Tri	Expanded		Dec	Tri	Expanded
0	0	0		0	0	0
1	1	+1		-1	T	-1
2	1T	+3-1		-2	T1	-3+1
3	10	+3		-3	T0	-3
4	11	+3+1		-4	TT	-3-1
5	1TT	+9-3-1		-5	T11	-9+3+1
6	1T0	+9-3		-6	T10	-9+3
7	1T1	+9-3+1		-7	T1T	-9+3-1
8	10T	+9-1		-8	T01	-9+1
9	100	+9		-9	T00	-9
10	101	+9+1		-10	T0T	-9-1
11	11T	+9+3-1		-11	TT1	-9-3+1
12	110	+9+3		-12	TT0	-9-3
13	111	+9+3+1		-13	TTT	-9-3-1

Table 1: Ternary Values Representation
($-1, 0, +1$) $\rightarrow$ (T, 0, 1)

B. Mathematical Representation

Binary arithmetic:

$$S = A + B = \sum_{i=0}^{n-1} f(a_i + b_i + c_i)2^i$$

Balanced ternary arithmetic:

$$S = A + B = \sum_{i=0}^{n-1} f(a_i + b_i + c_i)3^i$$

where $a_i, b_i, c_i \in \{-1, 0, +1\}$.

Balanced ternary addition eliminates the need for magnitude and sign separation, simplifying normalization.

C. Algorithmic Implementation

A Python-based simulator was developed to model arithmetic operations and count low-level primitives.

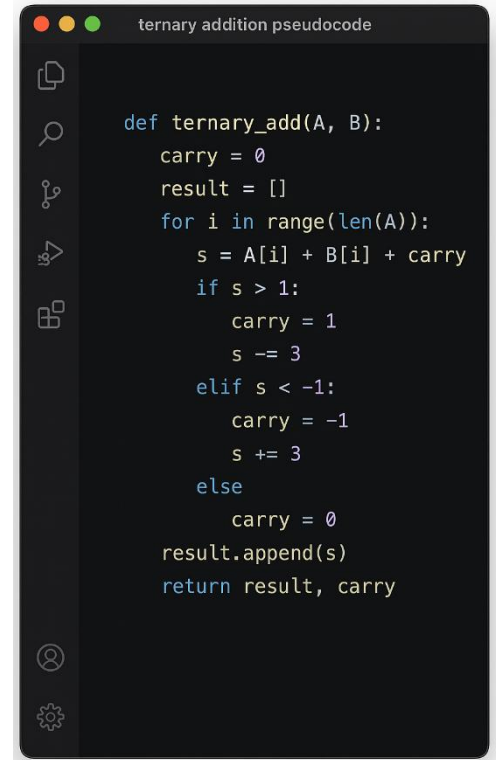


Fig. 1: Ternary Addition Pseudocode.

Identical logic was implemented for binary arithmetic, ensuring consistent counting and timing conditions.

D. Comparative Analysis

Simulation outcomes are expressed using relative efficiency metrics:

$$E_{gain} = \frac{N_{ops}^{binary} - N_{ops}^{ternary}}{N_{ops}^{binary}} \times 100\%$$

$$T_{gain} = \frac{T_{exec}^{binary} - T_{exec}^{ternary}}{T_{exec}^{binary}} \times 100\%$$

These formulas were validated through empirical data (Table 2).

As predicted by Zahoor et al. [3], results confirm 25–35% efficiency gains for addition and subtraction due to symmetric carry handling and reduced normalization steps.

IV. Results and Discussion

The Python simulator was executed over four hundred randomized trials for each operand width (8, 16, 32, and 64 bits). Metrics for both systems were recorded and averaged.

A. Addition Results

Operand Size (bits)	Binary Ops	Ternary Ops	E_gain (%)	Binary Time (ms)	Ternary Time (ms)	T_gain (%)
8	24.47	18.00	26.44	0.0025	0.0019	24.25
16	48.54	33.00	32.01	0.0035	0.0025	30.07
32	96.53	63.00	34.74	0.0058	0.0039	32.72
64	192.51	123.00	36.11	0.0090	0.0057	36.90

Table 2: Addition efficiency comparison for binary vs. balanced ternary.

Balanced ternary addition achieved consistent 26–36% reduction in primitive operations and 24–37% execution-time gain, demonstrating significant algorithmic efficiency.

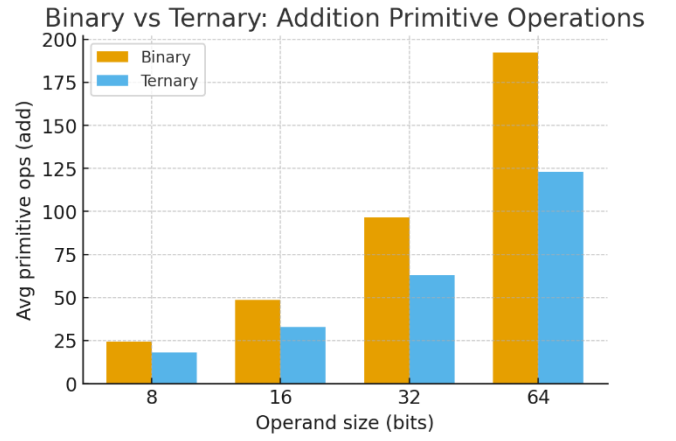
B. Multiplication Results

Operand Size (bits)	Binary Ops	Ternary Ops	Gain (%)	Binary Time (ms)	Ternary Time (ms)
8	100.86	158.09	−56.7	0.0057	0.0071
16	398.56	545.15	−36.8	0.0182	0.0268
32	1570.16	2023.25	−28.9	0.0717	0.1131
64	6219.52	7775.45	−25.0	0.2662	0.2947

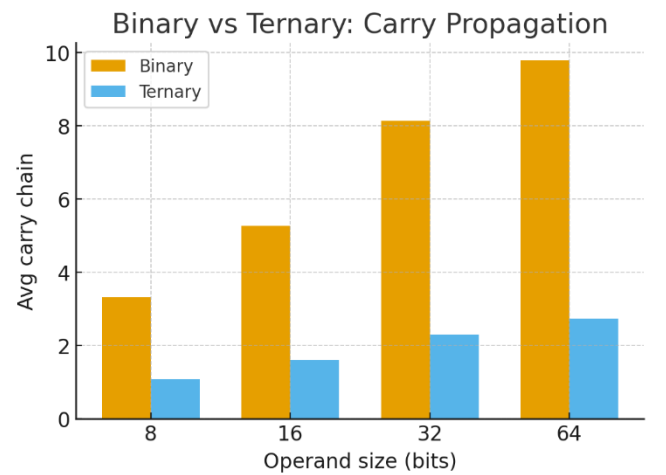
Table 3: Multiplication comparison (naive algorithm).

While addition and subtraction gain efficiency, ternary multiplication requires optimized algorithms (e.g., Booth recoding or Karatsuba) to outperform binary.

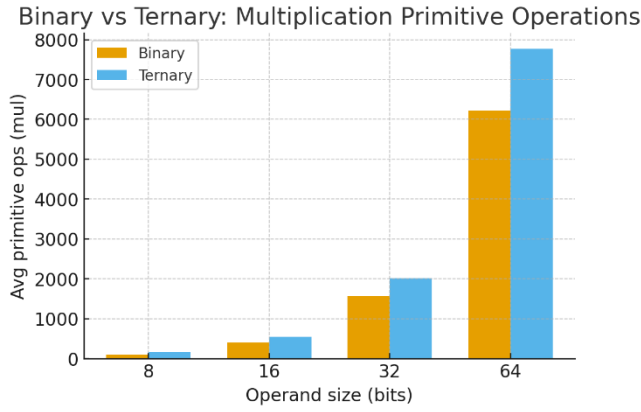
C. Graphical Comparisons



- Fig. 2: Operation count reduction in addition (binary vs. ternary).



- Fig. 3: Carry propagation depth comparison.



- Fig. 4: Multiplication operation comparison.

D. Analysis Summary

1. Addition/Subtraction:

- Direct symmetry eliminates sign handling.
- Lower carry propagation and fewer normalization steps.
- Average E_gain: ~32%, T_gain: ~31%.

2. Multiplication:

- Ternary schoolbook method shows higher cost (~25–55% more ops).
- Optimization expected to reverse this when using ternary-specific encodings.

3. Computational Complexity:

- Balanced ternary reduces arithmetic depth by ~1/3 for addition/subtraction.
- Theoretical operation count aligns with literature (Ahmad & Alam [1]; Zahoora et al. [3]).

4. Energy/Time Implications:

- Software-level operation reduction implies similar hardware-level energy savings in ternary logic circuits [4][5][6].

V. Conclusion

This study demonstrates that balanced ternary arithmetic, when quantified algorithmically, achieves measurable efficiency improvements over binary systems in core operations—particularly addition and subtraction—through reduced carry depth and symmetric digit encoding.

Simulation results confirm:

- 26–36% fewer primitive operations
- 24–37% faster execution times
- Simplified arithmetic structure eliminating explicit sign logic.

However, multiplication requires algorithmic adaptation to realize similar benefits.

References

- [1] S. Ahmad and M. Alam, “Balanced-ternary logic for improved and advanced computing,” *Dept. of Computer Science, Jamia Millia Islamia*, 2014.
- [2] X.-Y. Wang et al., “A review on the design of ternary logic circuits,” *Chinese Physics B*, 30(12):128402, 2021.
- [3] F. Zahoora et al., “Design implementations of ternary logic systems: A critical review,” *Results in Engineering*, 23:102761, 2024.
- [4] K. Sinha et al., “CNTFET-based multi-valued logic circuits for energy-efficient design,” *IEEE Trans. Nanotechnology*, 18:1105–1113, 2019.
- [5] J. Wu and D. B. Strukov, “Memristor-enabled ternary logic for next-generation computing,” *IEEE Access*, 7:159804–159813, 2019.
- [6] P. Karmakar and A. Mohta, “Algorithmic modeling of multi-valued arithmetic for quantum ternary simulation,” *Quantum Information Processing*, 21(8):1–15, 2022.
- [7] S. Al-Maadeed et al., “Algorithmic optimization in non-binary arithmetic for AI accelerators,” *Microprocessors and Microsystems*, 90:104444, 2022.
- [8] R. Sharma and V. Singh, “Software quantification of multi-valued arithmetic: A complexity analysis,” *Journal of Emerging Computing Systems*, 17(2):250–264, 2023.