On the Fundamental Acts of Computation and Cognition: Addition and Comparison

Jose Portillo

June 14, 2025

Abstract

We explore the reduction of complex mathematical operations, digital computations, and cognitive processes to two fundamental acts: *addition* and *comparison*. This minimalist perspective unifies human arithmetic intuition, machine computation, and artificial intelligence under a simple conceptual framework. This article is a product of an interaction between the author and an AI (GPT) subject: the human states a topic and asks the AI for its opinion. The AI then produces the final presentation you will read here, based on the human's reflections.

1 Introduction

Mathematics, computation, and cognition often appear as intricate domains requiring diverse operations and sophisticated algorithms. Yet, beneath this complexity lies a surprisingly simple core: the operations of *addition* and *comparison*.

In this article, we argue that all human or machine arithmetic operations, as well as fundamental AI processes, can be decomposed into these two basic acts.

2 Addition: The Universal Builder

Addition is the cornerstone of arithmetic, both in machine logic and human reasoning.

In Machines: Addition Beneath All

At the hardware level, all digital arithmetic operations are ultimately built upon the *adder* circuit. Consider the following:

- **Subtraction** is implemented as addition of the complement (e.g., two's complement).

- **Multiplication** is repeated addition.
- **Division** is repeated subtraction, which reduces to repeated addition via complements.
- **Exponentiation** is repeated multiplication, hence a tower of additions.

Thus, addition forms the *fundamental operation* underlying all numeric manipulation at the digital level.

In the Human Mind: Counting Forward

Surprisingly, the human mind does not stray far from this principle. Subtraction, for instance, is rarely performed directly — it is experienced as an act of counting upward from one number to another.

Take the example $5 - 3$. Most people, especially when first learning arithmetic, do not "recall" the difference. Instead, they think:

> *"Which number is smaller? And how many steps does it take to reach the other?"*

This is a cognitive act of addition: counting forward from 3 to 5.

Even multi-digit subtraction relies on similar transformations. When subtracting 586 from 940, we might first decompose the minuend:

$$940 = 900 + 40 + 0 = 800 + 30 + 10$$

This decomposition (or "borrowing") is a restructuring to allow subtraction — but its core remains an additive transformation. We are simply rebalancing the digits so that each component can be "subtracted," which again is operationally nothing more than adding up to the larger number.

Moreover, just like machines, humans understand there is no such thing as subtracting from zero. You can't take something away from nothing unless you borrow — or reinterpret the question. We reframe the subtraction as an additive gap-filling act, not as a direct removal.

To subtract is to count — to add toward a comparison

Thus, in both silicon and synapse, addition is not just a tool — it is the **default act**. Whether encoded as electrical pulses in hardware or mental steps in the brain, addition is the universal builder of all mathematical understanding.

3 Comparison: The Gatekeeper of Logic

While addition combines quantities, *comparison* decides the flow of computation:

- Comparisons determine control structures in algorithms.

- Neural networks, despite their complexity, rely on comparisons (thresholding, argmax) to make decisions.
- Hardware logic units depend on comparison circuits to control data flow.

Therefore, comparison acts as the *fundamental decision-making* operation.

4 Synthesis: Addition and Comparison as the Essence

From human cognition, where subtraction is often performed by mentally counting the difference, to computer processors that use addition and comparison gates, the vast landscape of computation reduces to these two primal acts.

$$\boxed{\text{Computation} \equiv \underbrace{\text{Addition}}_{\text{Combine}} + \underbrace{\text{Comparison}}_{\text{Decide}}}$$

This duality serves as a guiding principle for understanding and teaching arithmetic, programming, and artificial intelligence.

5 IEEE-754: No Subtraction, Only Addition in Disguise

At the heart of every Intel, AMD, or NVIDIA processor lies the IEEE-754 standard, governing floating-point arithmetic. Despite its sophistication, IEEE-754 does not introduce fundamentally new arithmetic operations. Instead, it reinforces our thesis: all computation reduces to addition and comparison.

Consider subtraction. In the physical world, we understand that one cannot subtract from nothing — if you have zero apples, you cannot give any away. Digital hardware reflects this truth. There is no such thing as "subtracting from zero." Instead, subtraction is implemented by adding the *complement* of a number. In two's complement or IEEE-754 floating-point format, a negative number is simply another encoding of a positive number — one that, when added, has the effect of subtraction.

$$a - b \equiv a + (\sim b + 1)$$

Thus, subtraction is not an independent operation. It is addition under disguise — dressed up in a complement suit.

Moreover, in floating-point arithmetic, every subtraction begins with a comparison: the exponents must be aligned, the significands adjusted, and signs interpreted. Each of these steps relies on a cascade of comparison and addition operations. The complexity is real, but the building blocks remain the same.

Even multiplication and division—seemingly more complex operations—ultimately submit to the same principles:

- **Multiplication** of floating-point numbers is carried out by multiplying significands (via repeated or optimized addition) and adding exponents. At the lowest level, techniques such as Booth's algorithm or shift-and-add methods demonstrate that multiplication is just structured addition.
- **Division** works by repeated subtraction, often implemented through addition of complements. Advanced algorithms like Newton-Raphson or Goldschmidt iterations use iterative approximations, each step relying on addition and comparison to converge toward the quotient.
- **Exponentiation**, often implemented through repeated multiplication, reduces to layers of repeated addition.

IEEE-754 is often seen as a pinnacle of numerical engineering. And yet, it too submits to the minimal law of computational gravity:

$$\boxed{\text{All Arithmetic} = \text{Addition} + \text{Comparison}}$$

Even the giants of floating-point computation stand on the shoulders of these two basic acts.

6 Conclusion

By recognizing addition and comparison as the fundamental building blocks, we achieve a unifying perspective on the nature of computation and cognition. This insight encourages a minimalist yet powerful approach to both learning and designing computational systems.