Advances in Nonlinear Variational Inequalities
ISSN: 1092-910X
Vol 26 No. 2 (2023)

Abacus Algorithms: A Pure Mathematical Approach to Ancient Calculation Tools

Sayed Ibrahim Ali[1], Khaled Elballah[2], Alaa Sayed[3], Mostafa Shaban[4]

[1] Department of Family and Community Medicine, College of Medicine, King Faisal University, Al-Ahsa, Saudi Arabia.

[2] Associate professor, College of Education, Department of Special Education, King Faisal University, Al-Ahsa, Saudi Arabia.

[3] Nursing Student, Faculty of Nursing, Helwan University, Egypt.

[4] Lecturer, Geriatric Nursing Department, Faculty of Nursing, Cairo University, Egypt.

Corresponding author: Sayed Ibrahim Ali email: seali@kfu.edu.sa

Article History:

Received: *15-04-2023*

Revised: *10-06-2023*

Accepted: *30-06-2023*

Abstract: The abacus is one of the oldest calculating tools still in use today. Despite its simplicity, the bead-based interface allows users to conduct complex mathematical operations through a system of sliding beads along wires or rods. While the physical abacus itself provides an intuitive and visual approach to calculation, the underlying operations rely on fundamental mathematical principles. This paper provides a comprehensive mathematical framework that formally describes the algorithms behind abacus calculations. Beginning with basic abacus configuration, we define key components like rods, beads, and bead values required to model abacus states. We then characterize the core abacus algorithms for addition, subtraction, multiplication, and division through set notation, recurrence relations, and state transition diagrams. Our formalized abacus algorithms leverage concepts from number theory, modular arithmetic, combinatorics, and algebra. In addition to offering new mathematical insights into ancient technologies, our work helps bridge connections between the tangible abacus interface and the abstract algorithms powering it. Through examples and proofs, we show how bead manipulations precisely correspond to mathematical transformations. This level of formalization not only helps explain the effectiveness of the abacus, but also illustrates how even rudimentary calculation tools utilize profound mathematical ideas. Our mathematical abacus framework lays the foundation for further analysis as well as modifications and extensions of the classic abacus approach.

Keywords: Abacus Algorithms; Calculation Tools; Mathematical Framework; Addition; Subtraction; Multiplication.

1. Introduction

The abacus is one of the oldest known calculating devices still in continuous use today, serving as an important mathematical aid for thousands of years across many cultures before the adoption of modern computing technologies[1]. At its core, the abacus provides a physical interface to conduct arithmetic through the manipulation of beads on a frame of rods or wires. Despite the simplicity of sliding beads back and forth along rods, the tool allows users to carry out complex calculations through practiced finger motions[2]. However, underneath the tangible bead interface lies a set of algorithms derived from key mathematical concepts that give structure to and formally describe abacus computations[3].

While the physical abacus provides an intuitive and visual approach to calculation through bead configurations representing numbers, the abstract algorithms powering the computations utilize important mathematical ideas like number systems, modular arithmetic, and state transformations [4]. Formalizing these algorithms through mathematical notation helps explain why the abacus works so effectively as well as how bead manipulations correspond to underlying mathematical operations[5]. Furthermore, a formal framework helps characterize relationships between abacus algorithms for different operations like addition, subtraction, multiplication, and division[6]. Finally, mathematical formalization enables new abacus algorithms to be developed, modifications and optimizations to existing methods, as well as connections with other mathematical domains. There is study conducted by the Department of Educational Psychology at the Faculty of Education, Helwan University in 2012. It aimed to prepare a program based on the abacus to understand its effectiveness in developing the mental and cognitive abilities of primary school students. The study included 68 fifth-grade students, divided into control and experimental groups, and involved pre- and post-tests using Raven's Progressive Matrices[7].

In this paper, we provide a comprehensive mathematical framework to model abacus computation by precisely specifying the algorithms for conducting arithmetic on an abacus. We begin by defining key components and variables to mathematically represent abacus states including rods, beads, and bead values. Building on this, we formally describe the core abacus algorithms for addition, subtraction, multiplication, and division through set notation, recurrence relations, transition diagrams, and other mathematical constructs[8]. Through rigorously defining the algorithms and demonstrating them through examples, we show how each step in a calculation procedure maps to precise mathematical transformations of abacus states[9]. Our formalized approach not only helps explain why the simple abacus tool can support complex math but also reveals deeper connections between tangible calculation interfaces and the abstract algorithms enabling them.

Abacus Structure and Configuration

Before defining abacus computation algorithms, we first establish notation for modeling abacus structure and representing abacus status. We define the key components as:

Rods: Horizontal parallel rods upon which beads are placed. Abaci generally contain between 9 to 15 rods. Beads: Beads placed on rods represent numerical units. Each rod contains between 5 to 7 beads. Bead Values: Each bead has an associated integral value based on rod position. Bead values on a rod increase from bottom to top. **Base**: The bottom bead on each rod represents the base value for that rod position. Subsequent beads represent multiples of the base value.

Using these components, we define an abacus state A as the collection of beads currently on the rods:

$A = \{B_1, B_2, ..., B_n\}$

Where Bi represents the $_i$th bead on the abacus. Note that multiple beads can share the same rod, but no two beads may occupy the same position on a single rod.

Each bead B_i has an associated value V_i based on its rod position. V_i is defined as:

$V_i = b_i * c_i$

Where: bi = base value for $_i$th rod c_i = vertical level of B_i (1 for bottom bead, 2 for next level up, etc.)

Given values assigned to each bead, we can calculate the total value V represented by an abacus state A as:

$V(A) = \Sigma V_i$ for $_i = 1$ to n

Where n is the number of beads in A. Computation algorithms covered in subsequent sections define state transitions that transform A by sliding beads vertically and horizontally between rods to effectively modify V.

Abacus Addition Algorithm

We now define algorithms for conducting addition on the abacus framework. Given two input numbers X and Y represented by abacus states A_x and A_y, the addition algorithm produces a new state A_z representing the summation Z = X + Y. We model the algorithm through the following steps:

1. Initialize $A_z = A_x$ This copies first addend state A_x to result state A_z.
2. For each bead B_i in A_y, move corresponding bead in A_z up by ci levels.
   This transfers beads from second addend A_y to the result representation A_z effectively summing the values. If a rod exceeds capacity after moving a bead, a carry procedure is invoked.
3. Repeat carry procedure until no overflows exist. The carry process transfers overflow from higher value rods to next rod, maintaining proper bead values.

We formalize steps 2 and 3 as follows:

$A_z(k+1) = A_z(k) + B_i$ if rod capacity not exceeded $= A_z(k) + B_i - c_i * Max(rod) + Max(rod) * b_j$ if overflow

Where: $A_z(k)$ is the abacus state A_z at iteration k B_i is the $_i$th bead from addend state A_y b_j is base value of next higher rod Max(rod) is capacity of current rod

To demonstrate, consider adding 6 + 4 on a 5 rod abacus. Following the algorithm:

1. A_z initialized to A_x with value 6
2. Transfer A_y beads valued 1 and 3 to A_z rods
3. Carry 1 bead from 1s to 10_s rod

The final state A_z represents 6 + 4 = 10 as expected.

Abacus Subtraction Algorithm

Subtraction relies on the complement principle [3] by taking the 9's complement of the subtrahend then applying the addition algorithm. This utilizes the fact that:

X - Y = X + (Max - Y) where Max is maximum abacus value

The algorithm operates as follows:

1. Initialize $A_z = A_x$
   Set up minuend in result
2. Complement beads from A_y by sliding each B_i down until bottom New beads are added to fill gaps between existing beads in A_y
3. Apply addition algorithm to add Ax and A'_y A'_y represents 9's complement of subtrahend A_y

Much like for addition, this transforms state A_z representing the difference X - Y through precise bead manipulations. A_s an example, subtracting 4 from 6:

1. A_x holds minuend value 6
2. Complement beads in Ay to add 5
3. Addition gives 6 + 5 = 11 Final bead on 1s rod represents 1 for difference of 6 - 4

Abacus Multiplication Algorithm

The multiplication procedure utilizes repeated addition based on the equation:

X * Y = X + X + ... + X (Y times)

We model this recursively through the following steps:

1. Initialize $A_z = \{0\}$ and Sum = A_x
2. For i = 1 to Y:
- Sum = Sum + A_x (add A_x into Sum i times)
- If overflow, carry beads
3. A_z = Sum Result state A_z now holds product value

This is formalized mathematically as:

$A_z(i+1) = Sum(i) + A_x$ if capacity permits $= Sum(i) + A_x - c_i*Max(rod) + Max(rod)*b_j$ otherwise

Where: A_z is result state after i iterations Sum(i) is cumulative sum state
A_x is multiplicand state

And initial conditions: $A_z(0) = \{0\}$ $Sum(0) = A_x$

The carry process is identical to that used in addition. An example multiplying 6 by 4:

1. A_x holds multiplicand 6; A_y has multiplier 4
2. Successively add A_x into Sum 4 times
3. Final state A_z represents product 6 * 4 = 24

Abacus Division Algorithm

The division algorithm makes use of repeated subtraction, following the basic definition:

X / Y = Z such that Y * Z = X

We compute integer quotient Z through the following recursive process:

1. Initialize: Q = 0 (Quotient counter) R = A_x (Remainder set to dividend)
   D = A_y (Divisor)
2. While R ≥ D: R = R - D (subtract D from R)
   Q = Q + 1 (increment quotient)
3. A_z = Q Final state A_z represents integer quotient result

This is expressed mathematically through the recurrence:

$R(k+1) = R(k) - D$ if $R(k) \geq D$ $Q(k+1) = Q(k) + 1$ ”

With initial values: $R(0) = A_x$
$Q(0) = 0$

And final result:
$A_z = Q(n)$

Where n iterations performed until remainder R < divisor D.

For example, to divide 30 by 6:

1. Set R = 30, D = 6, Q = 0

2. Subtract D = 6 from R until < 6
3. Perform 5 subtractions so Q = 5
4. Quotient state A_z represents 30 / 6 = 5

This extends directly to fractional quotients through tracking of remainders.

2. Discussion

By rigorously defining abacus states and algorithms through mathematical constructs like sets, relations, diagrams, and examples, we have provided a comprehensive framework for analyzing the fundamental basis by which abaci perform calculations. While the physical abacus interface centered around tangible bead manipulations gives the impression of simplicity, formalizing the underlying processes illustrates deep connections with important mathematical theories[1]. Our abacus state representations draw from ideas in combinatorial math and number systems while the addition, subtraction, multiplication, and division algorithms leverage key concepts from number theory, algebraic structures, and state machine modeling[10].

Beyond explaining the effectiveness of abacus calculation, our formalized approach helps relate the physical computational interface provided by abacus rods and beads with the abstract mathematical processes driving the computations[11]. Furthermore, explicitly defining algorithms through notation, equations, recursive relations, and transition diagrams allows modifications and extensions to be readily incorporated within the established framework[12]. This facilitates continued evolution of the simple yet powerful abacus approach to achieve higher efficiency and accuracy as well as adaptations to different number bases[13]. Additionally, analysis of time complexity, optimal configurations, and comparisons against other calculation methods is enabled by mathematically defining the algorithms[14].

While we have covered core arithmetic operations, many opportunities exist for further formal exploration of abacus computation. Extending to negative numbers, decimals, mod operations, set operations, and square roots could reveal additional useful mathematical connections[15]. Investigating mechanical abacus implementations or abacus software could also benefit from algorithmic modeling[16]. Ultimately, our mathematically-grounded abacus algorithms help bridge connections between one of the most ancient calculation tools and the sophisticated abstractions enabling efficient, precise, and advanced computation[17].

3. Conclusion

In this work, we have presented a comprehensive mathematical framework to formally model the algorithms underlying abacus computation. By rigorously defining key components like rods, beads, and bead values, we constructed an abacus state representation using set notation. Building on this, we specified algorithms for addition, subtraction, multiplication, and division through mathematical relations, transition diagrams, and examples. Our formalized approach reveals the sophisticated mathematical ideas like number systems, modular arithmetic, and state transformations that enable the simple physical abacus interface to support complex calculations.

The main benefits provided by mathematically formalizing abacus algorithms are three-fold. First, it allows us to precisely explain the effectiveness of the abacus as a calculation tool by bridging connections between physical bead manipulations and the abstract mathematical processes driving them. Second, explicitly defined algorithms can be readily analyzed, optimized, modified, and extended within the established framework. Finally, formalization enables connections with computer science topics like complexity analysis, mechanical computation, and new algorithm design.

While we focused on modeling standard arithmetic operations, many avenues exist for further applying our abacus framework. Possible directions include extending algorithms to negative numbers, decimals, and additional mathematical functions as well as comparing abacus computation with other calculation tools. There is also opportunity to leverage the modular and visual nature of the abacus for novel applications in education, humancomputer interaction, and recreational math. Ultimately, by interfacing ancient tools with modern mathematical formalism, new insights can be gained into culturally significant computation techniques that continue to inform the design of new technologies.

Funding: This research was funded by Deanship of Scientific Research at King Faisal University, Saudi Arabia (GRANT5,379).

Acknowledgments: The authors acknowledge the Deanship of Scientific Research at King Faisal University for obtaining financial support for research, authorship, and the publication. of research (GRANT5,379)

Conflicts of Interest: The author declares no conflict of interest

References:

[1] Wang C: A Review of the Effects of Abacus Training on Cognitive Functions and Neural Systems in Humans. Front Neurosci. 2020, 14:. 10.3389/fnins.2020.00913

[2] Lima-Silva TB, Barbosa ME de C, Zumkeller MG, et al.: Cognitive training using the abacus: a literature review study on the benefits for different age groups. Dement Neuropsychol. 2021, 15:256–66. 10.1590/1980-57642021dn15-020014

[3] Xiong P, Hu X, Huang B, Zhang J, Chen Q, Liu H: Increasing the efficiency and accuracy of the ABACUS protein sequence design method. Bioinformatics. 2020, 36:136–44. 10.1093/bioinformatics/btz515

[4] Chen F, Hu Z, Zhao X, Wang R, Yang Z, Wang X, Tang X: Neural correlates of serial abacus mental calculation in children: A functional MRI study. Neurosci Lett. 2006, 403:46–51. 10.1016/j.neulet.2006.04.041

[5] Wang C-C, Sun Y-C, Zhang Y, Xin Y-W, Gao W-X, Jia L-P, Liu X-Q: An ERP study on the influence of mental abacus calculation on subthreshold arithmetic priming in children. Neuroreport. 2022, 33:116–28. 10.1097/WNR.0000000000001761

[6] Burke PC, Eichmuller L, Messam M, Duffy DA, Borkowski RG, Gordon SM, Fraser TG: Beyond the abacus: Leveraging the electronic medical record for central line day surveillance. Am J Infect Control. 2019, 47:1397–9. 10.1016/j.ajic.2019.05.013

[7] Ali SI: The effectiveness of a program using the abacus in developing Mental abilities of primary school students. 2012.

[8] Szabados B, Kockx M, Assaf ZJ, et al.: Final Results of Neoadjuvant Atezolizumab in Cisplatin-ineligible Patients with Muscle-invasive Urothelial Cancer of the Bladder. Eur Urol. 2022, 82:212–22. 10.1016/j.eururo.2022.04.013

[9] Srinivasan M, Wagner K, Frank MC, Barner D: The Role of Design and Training in Artifact Expertise: The Case of the Abacus and Visual Attention. Cogn Sci. 2018, 42:757–82. 10.1111/cogs.12611

[10] Dong S, Wang C, Xie Y, Hu Y, Weng J, Chen F: The impact of abacus training on working memory and underlying neural correlates in young adults. Neuroscience. 2016, 332:181–90. 10.1016/j.neuroscience.2016.06.051

[11] Du F, Chen F, Li Y, Hu Y, Tian M, Zhang H: Abacus Training Modulates the Neural Correlates of Exact and Approximate Calculations in Chinese Children: An fMRI Study. Biomed Res Int. 2013, 2013:1–12. 10.1155/2013/694075

[12] Xie Y, Weng J, Wang C, Xu T, Peng X, Chen F: The impact of long-term abacus training on modular properties of functional brain network. Neuroimage. 2018, 183:811–7. 10.1016/j.neuroimage.2018.08.057

[13] Brooks NB, Barner D, Frank M, Goldin-Meadow S: The Role of Gesture in Supporting Mental Representations: The Case of Mental Abacus Arithmetic. Cogn Sci. 2018, 42:554–75. 10.1111/cogs.12527

[14] Li Y, Chen F, Huang W: Neural Plasticity following Abacus Training in Humans: A Review and Future Directions. Neural Plast. 2016, 2016:1–9. 10.1155/2016/1213723

[15] Cho PS, So WC: A Feel for Numbers: The Changing Role of Gesture in Manipulating the Mental Representation of an Abacus Among Children at Different Skill Levels. Front Psychol. 2018, 9:. 10.3389/fpsyg.2018.01267

[16] Cortesi M, Llamosas E, Henry CE, Kumaran R-YA, Ng B, Youkhana J, Ford CE: I-AbACUS: a Reliable Software Tool for the Semi-Automatic Analysis of Invasion and Migration Transwell Assays. Sci Rep. 2018, 8:3814. 10.1038/s41598-018-22091-5

[17] Barber B, Scholes M: Reflections On the Development of Medical Informatics. Acta Inform Medica. 2014, 22:18. 10.5455/aim.2014.22.18-24