

Arithmetic Algorithms Taught in Schools

Joint Mathematics Meetings
Baltimore
January 15-18, 2014

Patricia Baggett
Dept. of Math Sci.
New Mexico State Univ.
Las Cruces, NM 88003-8001
baggett@nmsu.edu

Andrzej Ehrenfeucht
Computer Science Dept.
University of Colorado
Boulder, CO 80309-0430

andrzej@cs.colorado.edu

<http://www.math.nmsu.edu/~breakingaway/>

Outline

1. A short historical introduction
2. John Napier's original counting board (from *Rabdologiæ*, 1617)
3. Modified boards and their uses
4. Final comments

1. A short historical introduction

The “traditional” computational algorithms for the basic operations have been taught in schools for a long while.

But why these particular algorithms, and not others?
Are they easier to learn and more efficient than other algorithms?

There is a myth that Hindu-Arabic notation and algorithms that are supported by it, namely, algorithms that are still taught in schools, replaced other techniques used in Europe, exactly because of this: because they were easier to learn and use, and they were more efficient.

But historical data give us a different picture.

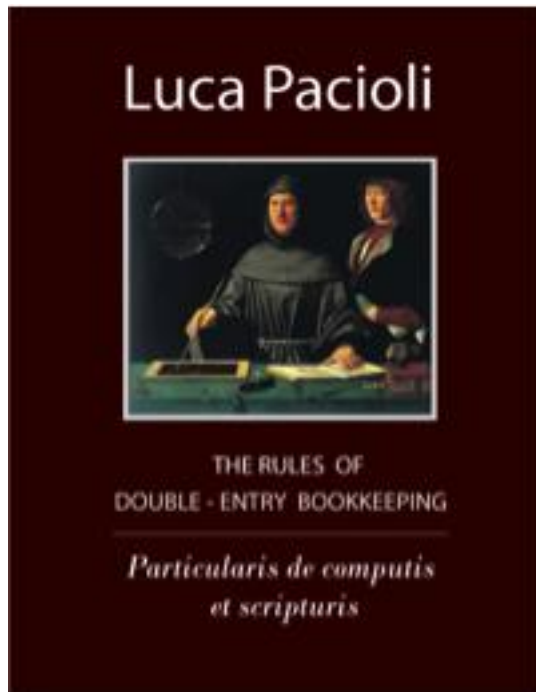
Hindu-Arabic arithmetic became known in Europe circa 1200 CE (Fibonacci, 1170-1250). Circa 1400 it began replacing previous methods of computation with jetons that were based on the Roman abacus,



and it became the dominant arithmetic by circa 1700.

There were two main factors that promoted this change:

1. The switch to trade based on *credit*, rather than on *the transfer of money*, by Italian banks. This switch was supported by double-entry bookkeeping (Pacioli, 1445-1517).



BOOK-KEEPING BY DOUBLE ENTRY;

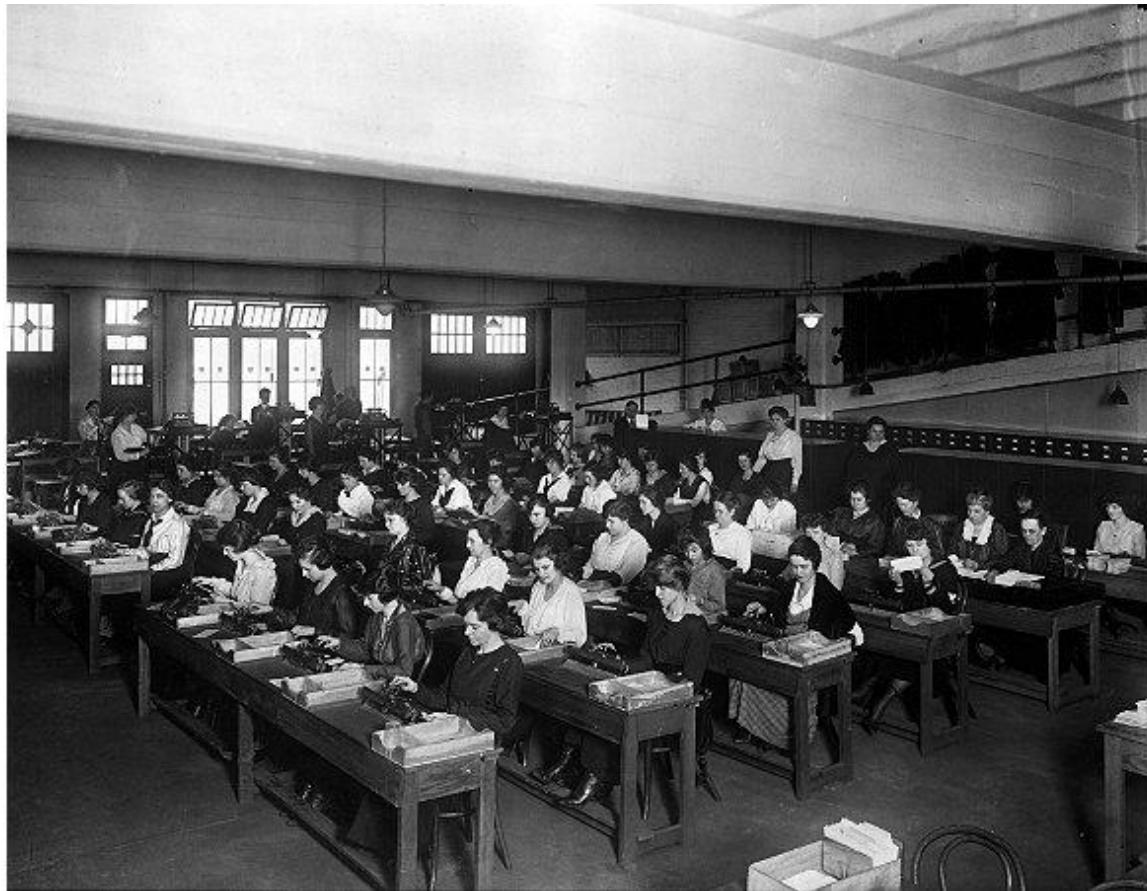
Or, according to the ITALIAN METHOD.

THIS method is said to be by Double Entry, because every article is twice entered in the Ledger, viz. on the Dr. side of one account, and on the Cr. side of another: and it is called the Italian method, because of its having been invented in Italy.

Hutton, 1807

In this method of bookkeeping, an arithmetic computation was part of a legal record of transactions, so it had to be done in writing (in ink), and it had to be protected from forgery.

Learning the written algorithms was, and is, time consuming because it requires memorizing addition and multiplication “facts”. But this new method also created the profession of “human computers”, namely, accountants whose only tasks were to do computations.



2. The introduction of decimal fractions (Stevin, 1548-1620), the invention of logarithms (Napier, 1550-1617), the construction of logarithmic tables and the slide rule (Oughtred, 1575-1660) replaced the use of the abacus in technical computations used by craftsmen, because craftsmen needed only approximate answers.

During the nineteenth century and the first two thirds of the twentieth century, public education required that students learn traditional algorithms during their first eight years of schooling, and that they learn logarithms during their last four years (if they progress that far).

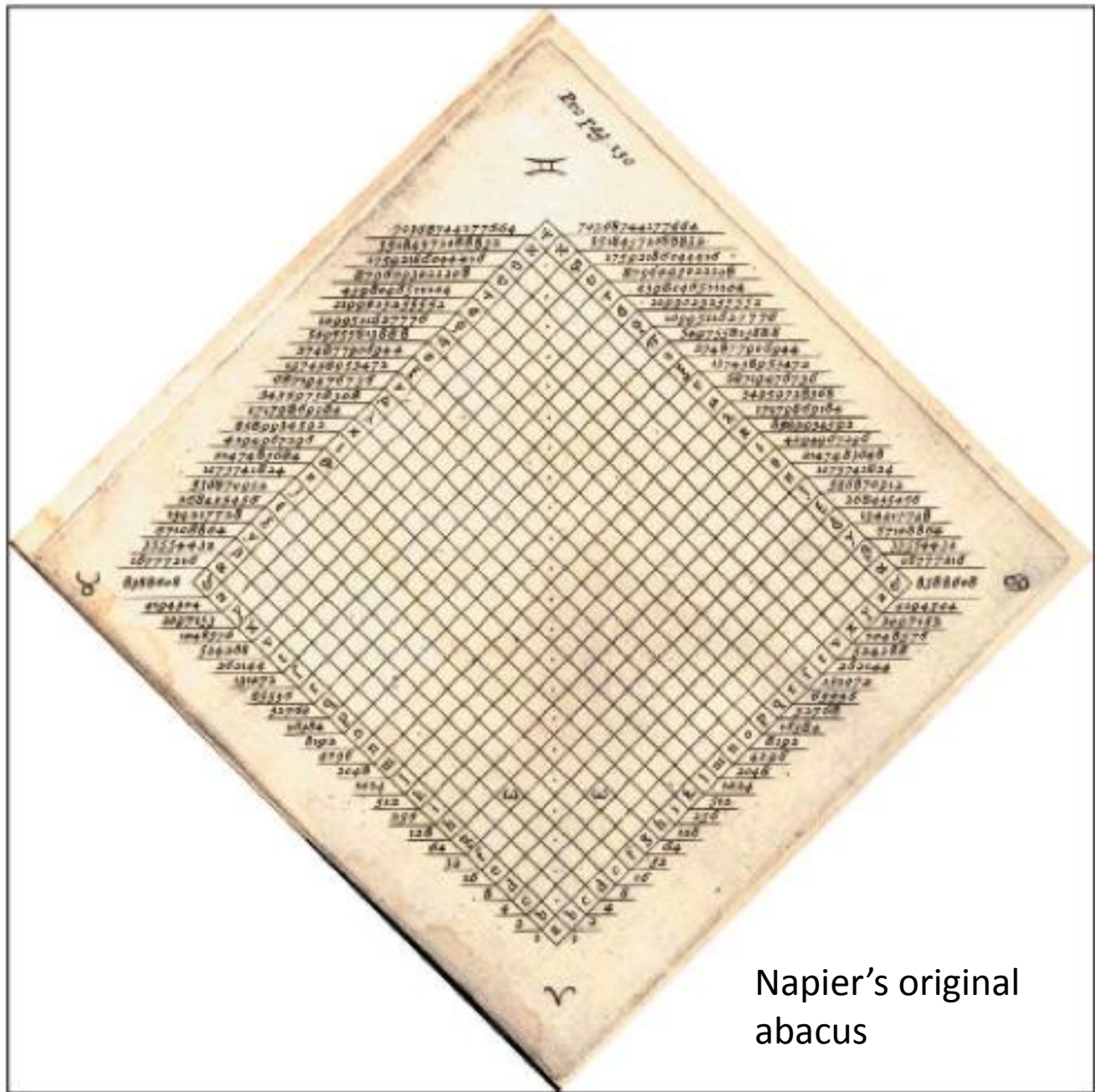
But the invention of computers made long-hand calculations obsolete. Thus, schools switched their focus from *skills* in arithmetic to *understanding* arithmetic. But they still require that students achieve at least some mastery of “standard algorithms”. This means that students spend hundreds of hours learning “arithmetic facts”.

We think that teaching different arithmetic algorithms, which do not require learning “facts” as a prerequisite, can provide better and faster understanding of arithmetic.

The algorithms that we suggest are carried out on a modified version of a counting board that was invented by John Napier and described in his 1617 *Rabdology*. His board was essentially ignored by everyone, with the exception of Martin Gardner (1914-2010), who in 1986 recognized its originality.

2. John Napier's original counting board (1617)





Napier's original abacus

2^{n*2^m}							2^m
			64	32	16	8	4
			32	16	8	4	2
2^n			16	8	4	2	1

Napier's counting board was an n by n "chess board". A whole number, which he called a "location value", was assigned to each square. Location values formed a geometric progression with a factor of 2 in each row and each column.

Napier did not use a base-two representation. He invented his own, where $a = 1$, $b = 2$, $c = 2^2$, $d = 2^3$, and so on. So, for example, $eca = 2^4 + 2^2 + 1 = 21$.

Showing this schematically, a token (jeton) put on a square had its local value:

		●	
64	32	16	8
			●
32	16	8	4
●			●
16	8	4	2
8	4	2	1

Here, the sum is 38.

The rules for “regrouping” were simple.

Two tokens on one square could be replaced by one token above or one token to the left.

Any token could be moved to another location on the same SW to NE diagonal.

Napier showed how to translate numbers from decimal notation to his binary notation and back.

He showed how to compute sums, differences, products, and quotients on his board.





Wikipedia suggests that with this board, he tried to show the uses of base-two notation. But this work is clearly a continuation of his work on logarithms, because multiplication on his board uses the same principle as multiplication on a slide rule, and the fact that he used geometric progression with a factor of 2 is not essential.

3. Modified boards and their uses

We have made three changes in the original design:

1. We kept the factor of 2 in the geometric progression in the columns, but we changed it to 5 in the rows. So now the SE to NW diagonal holds powers of 10, and thus all computations can be carried out in base 10.
2. We allow negative exponents, n and m , for local values $2^n 5^m$. So we can represent any finite decimal fraction.
3. We use two-color tokens (white and red). Red tokens represent negative numbers. This allows us to represent negative numbers, and reduces subtraction to “adding the opposite”.

$2^n 5^n$							$2^n 5^{-n}$
	50	10	2	.4	.08		
	25	5	1	.2	.04		
	5	2.5	.5	.1	.02		
	2.5	1.25	.25	.05	.01		
$2^{-n} 5^{-n}$							$2^{-n} 5^{-n}$

				
	10			
	25	5	1	
	.2	.04		
	12.5		.5	.1
	2.5			

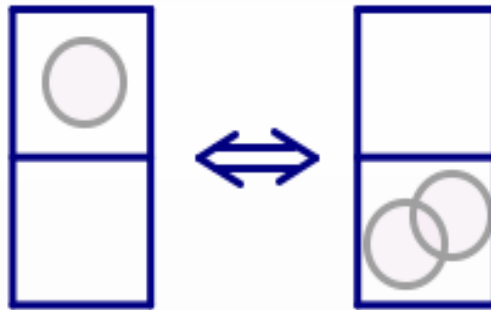
$$-10 + 25 + -.2 + 2.5 = 17.3$$

There are three rules for regrouping tokens:

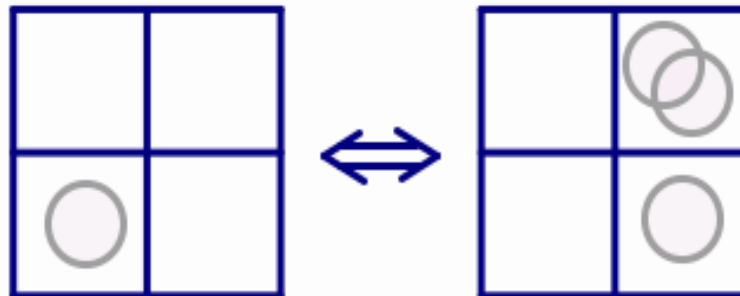
Rule 1. Two tokens having different colors can be put on any location or removed from it. This corresponds to the equality $x + -x = 0$.



Rule 2. A token can be exchanged for two tokens of the same color, at a square below it. This corresponds to $2x = x + x$.



Rule 3. A token can be exchanged for three tokens, one to the right, and two to the right and up. This corresponds to $5x = x + 2*(2x)$.



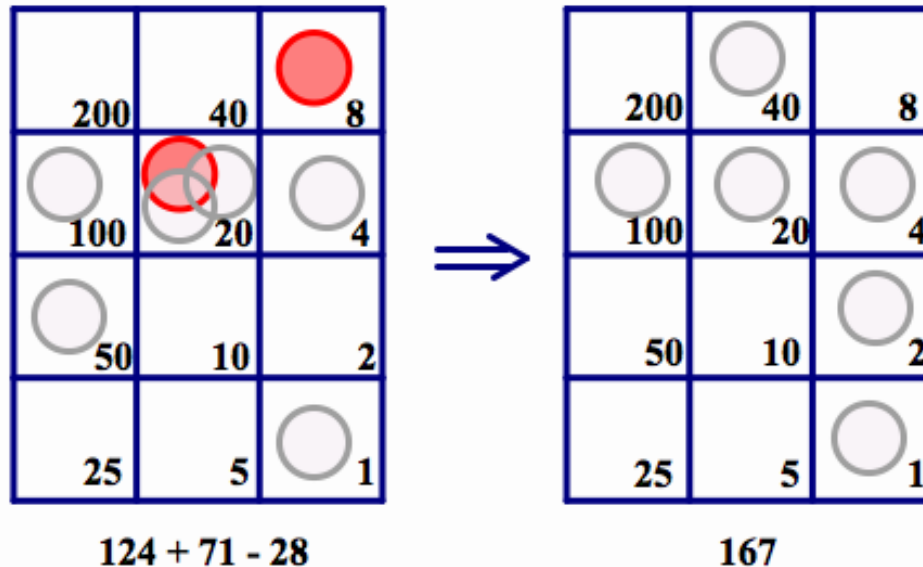
On this board, a large number of different arithmetic algorithms (including the “standard” ones) can be executed. We briefly describe here only three.

Addition and subtraction

Any one-digit decimal number requires only one or two tokens. So adding a list of positive and negative decimals can be done like this:

Put all the numbers on the board, and use the rules to regroup so that the tokens in one column represent a one-digit decimal.

Example: $124 + 71 - 28 = 167$

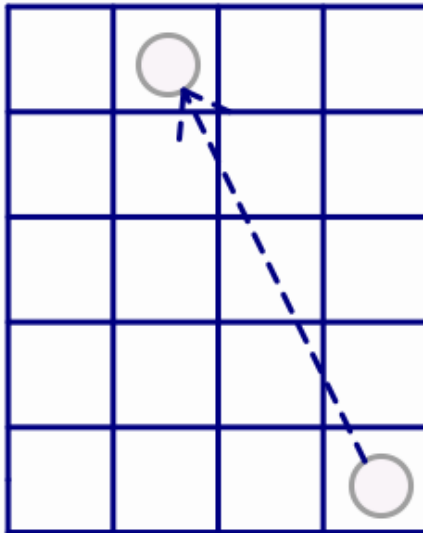


All the algorithms that we describe here are *flexible*. The method of regrouping is always left to the user. But every flexible algorithm can be replaced by a *rigid algorithm*, which would exactly specify each step in the computation.

Multiplication

There are many different algorithms for multiplication. We describe one that is different from the “standard” one and from Napier’s original algorithm.

Multiplication of a number x , represented by tokens on the board, by $2^n 5^m$ (where n and m are positive) is done by shifting all tokens of x n locations up and m locations to the left. So, in order to multiply x by y , we create one copy of x , shifted by $2^n 5^m$ corresponding to each token of y , and we add all these copies.



Multiplication by $400 = 2^4 \cdot 5^2$
(Move token up 4 and to the left 2.)

Division

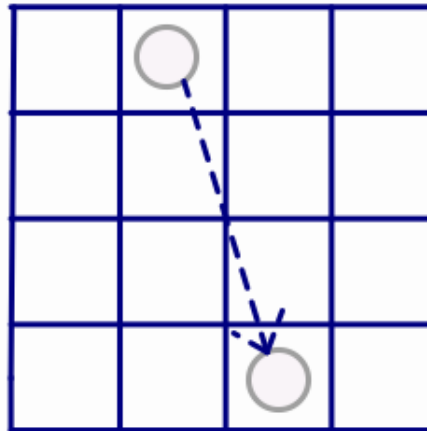
There are two concepts of division of finite decimals: Division with remainder, and approximate division in which the result is rounded to provide any specific accuracy. We show Brahmagupta's (597-668 CE) algorithm for approximate division.

In order to compute x/y , find z and $t = 2^n 5^m$, such that $y*z = 2^n 5^m - 1$. Thus,

$$x/y = (x*z)/(y*z) = x*z/(2^n 5^m - 1) = x*z*t^{-1} + x*z*t^{-2} + x*z*t^{-3} + \dots$$

and each term of this series is just a copy of $x*z$ shifted down and to the right!

To divide x by 13:
$$\frac{x}{13} = \frac{x*3}{40-1} = \frac{x*3}{40} + \frac{x*3}{40^2} + \frac{x*3}{40^3} + \frac{x*3}{40^4} + \dots$$



Division by $40 = 2^3 * 5$

(Move token 3 down and 1 to the right.)

4. Final comments

We think that using modified Napier boards of different sizes, depending on the level of children's knowledge, has several advantages for teaching arithmetic:

- a. The skills needed to use the boards are limited to reading and writing decimals, and to good mental skills of working with whole numbers smaller than or equal to 12. So using such a board *doesn't require much memorization*.
- b. The board provides one framework for whole numbers, integers, and (finite) decimal fractions. So these numbers are handled by *the same set of simple rules*.
- c. Many *different algorithms* can be carried out on these boards.
- d. The algorithms executed on these boards are *flexible*. The details of regrouping are left to the users, who have to *think what they are doing*, instead of following a memorized sequence of moves.
- e. Implementation of the algorithms is *efficient*. It allows users to carry out complex and *interesting* computations.

If you would like more details:

baggett@nmsu.edu

andrzej.ehrenfeucht@cs.colorado.edu

Thank you!