Matteo Pasquinelli

From algorism to algorithm: A brief history of calculation from the Middle Ages to the present day

This text, written by a professor of Media Philosophy at the University of Karlsruhe, Germany, and research group coordinator into Artificial Intelligence, traces the history of algorithmic processes and shows that its history started well before the so-called Information Age. The writer discusses binary codes, algorithms, and digital calculus among other areas and concepts that have now become familiar to many people, despite the fact that only a few understand what they represent and mean.

FROM ALGORISM TO ALGORITHM: A BRIEF HISTORY OF CALCULATION FROM THE MIDDLE AGES TO THE PRESENT DAY

Matteo Pasquinelli

Computer science defines the algorithm as a *step-by-step instructional procedure for transforming an input into an output independently of data and in the fewest possible steps.* But as the French mathematician Jean-Luc Chabert has pointed out, similar procedures have existed since the most ancient times, representing a way of thinking and problem solving that is not exclusive to the computer sciences, but is part of the cultural techniques of all human civilisations:

> Algorithms have been around since the beginning of time and existed well before a special word had been coined to describe them. Algorithms are simply a set of step by step instructions, to be carried out quite mechanically, so as to achieve some desired result [...]. The Babylonians used them for deciding points of law, Latin teachers used them to get the grammar right, and they have been used in all cultures for predicting the future, for deciding medical treatment, or for preparing food. Everybody today uses algorithms of one sort or another, often unconsciously, when following a recipe, using a knitting pattern or operating household gadgets.[1]

In other words, before its logical form was codified and mechanised as part of the information revolution, the algorithm – or more precisely algorithmic practices and thought processes – already existed. How did algorithms come to be automated and transformed into abstractions that are now completely incomprehensible to us?

This very brief essay proposes that the algorithm and the number have a joint history. It will show how these two notions have always been implicitly connected and how algorithmic practices are probably even older than mathematics itself. Specifically, the evolution of the idea of the algorithm will be reconstructed in relation to numeral systems from the medieval period to the present day, beginning with a linguistic coincidence that indicates deeper historical processes. Just as the medieval term *algorismus* signalled the passage from an additive (Roman) numeral system to a positional (Indo-Arabic) system, the contemporary term *algorithm* signalled the passage from a decimal system to a binary one. This happened not only for cultural and technological reasons, but also for fundamentally economic reasons, marking two transitions: in the first, from medieval society to commercial capitalism, and in the second, from industrial capitalism to the information society.

ALGORISMS AND MANUAL CALCULATION

In medieval Latin, the term *algorismus* referred to the manual procedure for calculating with Indo-Arabic positional numbers, which had arrived from the Middle East and were gradually supplanting Roman additive numbers. Positional numbers define quantities in progressive powers of ten, while

1. Chabert, Jean-Luc, ed., *A History of Algorithms: from the Pebble to the Microchip*, trans. Chris Weeks. Berlin and Heidelberg: Springer-Verlag 1999, p.1. Originally published in French as *Histoire d'algorithmes. Du Caillou à la puce*. Paris: Editions Belin, 1994.

"The evolution of the idea of the algorithm will be reconstructed in relation to numeral systems from the medieval period to the present day."

additive numbers do so simply by adding numbers, one next to the other (for example, the Indo-Arabic number '233' is equivalent to the Roman 'CCXXXIII'). As we know, Roman numbers are useful for dates and lists, but are a headache when it comes to simple mathematical operations, especially division. In the medieval period, Roman numbers were counted with an abacus, while with Indo-Arabic numbers, operations could be performed with pen and paper, following the new *algorisms*. The Italian merchant and mathematician Leonardo Fibonacci helped to introduce the Indo-Arabic system into Europe with his *Liber Abaci* of 1202, although this did not use the term *algorismus* but described the technique as *modus indorum*. The term *algorismus* appears in the 1240 poem *Carmen de Algorismo* by Alexandre de Villedieu, a manual for memorising the new calculation methods, composed in rhyming verses. A book printed in Venice in 1501, attributed to the thirteenth century monk Johannes de Sacrobosco, has the rather more aspirational title *Algorismus Domini* and uses diagrams to explain the new method of calculation with positional numbers.

It has only recently been established that the medieval term *algorismus* is a Latinisation of the name of the Persian scholar Muhammad ibn Musa al-Khwarizmi, a librarian from the House of Wisdom in Baghdad and author of a book on calculating with Indian numerals, written in about 825. Al-Khwarizmi's original manuscript in Arabic is lost, but from the twelfth century there were at least four Latin translations in circulation with different titles: one manuscript held by Cambridge University has the incipit 'DIXIT algorizmi' ('so said Al--Khwarizmi'), which the Italian mathematician Baldassarre Boncompagni translated as *Algoritmi de numero Indorum* in 1857.

The passage from the additive to the positional numeral system was not only a question of method; it also had a social and economic aspect, linked to the rapid increase in commercial trade across Europe and the Mediterranean. The positional system allowed numbers to be written down more concisely, thus speeding up financial accounting. In Italy, Florentine and Venetian merchants were the first to adopt Indo-Arabic numbers, which, as capital expanded, were preferred for their greater versatility in commercial transactions and in the management of figures. A drawing in the book *Margarita Philosophica* (1503), edited by the German monk and polymath Gregor Reisch, illustrates a dispute between the *abacists* (still using the Roman additive system and the abacus) and the new *algorists* (who had adopted the Indo-Arabic system and its algorisms and used a pen to make their calculations). The allegorical figure of *Arithmetica* oversees the dispute and clearly favours the algorist: she is wearing a robe decorated with the new numerals.

NAME

Previous pages:
Untitled, 1966

"In medieval Latin, the term *algorismus* referred to the manual procedure for calculating with Indo--Arabic positional numbers, which had arrived from the Middle East and were gradually supplanting Roman additive numbers."

Later, the term 'algorithm' was adopted by highly cultured European scholars such as Leibniz, who used it to define his method of differential calculus. In D'Alembert's *Encyclopédie*, the algorithm is broadly defined as:

> [...] an Arab term, used by various writers, and especially the Spanish, to mean the practice of algebra. It is also sometimes taken to mean arithmetic with figures... The same word is generally used to mean the method and notation of all types of calculation. In this sense, we say the algorithm of integral calculus, the algorithm of exponential calculus, the algorithm of sines, etc.

The methods for calculating by hand that are still taught in schools today are no more than the Indo-Arabic algorisms for the manipulation of numerical symbols. These methods have a recursive structure that can handle infinite and approximate values, as occurs for example in the simple division of numbers: 2/3 = 0.666666666... The simple continuous form of this fraction demonstrates that some numbers cannot be calculated and *represented* without the help of an algorithmic process. But at this point we could say that the very way numbers are written in a numerical system constitutes an algorithm. For example, the number 101 in Indo-Arabic numerals can be interpreted as follows:

> Take a linear sequence of positions that are to be occupied with symbols of quantity written from right to left. Each position progressively represents a power of ten and can be filled with one of the ten units: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. The first position represents normal units, the second position represents ten to the power of one (tens) the third represents ten to the power of two (hundreds) and so on. The value of a number represented in this way is produced by the addition of each unit after it has been multiplied by the power of ten represented by the position it occupies. When numbers are expressed in this way, the scale of powers of ten is not explicitly stated but remains implicit.

The number 101 thus equates to: (1 × 100) + (0 × 10) + (1 × 1). This explanation of the decimal system can easily be adapted to represent the binary system simply

by replacing the power of ten with the power of two, i.e. changing just one rule in the general numbering algorithm. In the binary system, the number 101 signifies a different quantity:

> Take a linear sequence of positions that are to be occupied with symbols of quantity written from right to left. Each position progressively represents a power of two and can be filled with one of the two ordinal units: 0 or 1. The first position represents two to the power of 0, i.e. normal units, the second position represents two to the power of one (two), the third represents two to the power of two (four) and so on. The value of a number represented in this way is produced by the addition of each unit after it has been multiplied by the power of two represented by the position it occupies. When numbers are expressed in this way, the scale of powers of two is not explicitly stated but remains implicit.

In this case, the number 101 equates to: $(1 \times 4) + (0 \times 2) + (1 \times 1)$. In both these paraphrases, the words of the natural language attempt to make visible the invisible matrix of numeral systems that are taught in schools mainly through exercises: rather than explaining the rules for constructing numbers, they codify them with step-by-step procedures.

The pedantry of these exercises enables something to be said that is not pedantic at all: all numeral systems are algorithmic in that they represent quantities by means of procedures (which can be highly codified or based on simple counting mechanisms). All numbers are algorithmic in that they are generated by numeral systems that are in fact algorithms.[2] In a manner of speaking, numbers do not count anything: they are simply signposts in sequences of quantities and procedures that often remain implicit. Thus, in an audacious dialectical turnaround, the algorithm no longer appears as a recent invention but as a structure of human thought and practice that goes back as far as the number itself. We might say that the algorithm and the number were separated at birth and that the former is only now beginning to be fully understood again, principally as a result of its automation on an industrial and social scale.

ALGORITHMS AND AUTOMATIC CALCULATION

Algorithms for manual calculation were mechanised and automated at different times and in different ways. In seventeenth century Europe, philosophers like Pascal and Leibniz were already designing manual calculators for performing the four basic operations, using the decimal system. In modern times, mathematical-abstract thought began to develop strictly in parallel with mechanical thought. Descartes' famous philosophical 'Method', for example, was notably 'mechanical' and 'algorithmic', with its emphasis on deconstructing a problem into its simplest elements. The Polish historian Henryk Grossmann maintained that it was not by chance that Descartes had come up with his rational method while he was himself designing machine tools. In particular, Grossmann added that 'every mathematical rule has this mechanical character that spares intellectual work and much calculation'.[3] Grossmann's two insights help us to under-

2. In mathematics and computer science, numerical algorithms refer to procedures for using numerical approximation (as opposed to symbolic manipulations) for problems of mathematical analysis. Here the term is hijacked to throw light on the constitution of numeration systems.
3. Henryk Grossmann, 'Descartes and the Social Origins of the Mechanistic Concept of the World', in *The Social and Economic Roots of the Scientific Revolution*, ed. Gideon Freudenthal and Peter McLaughlin. Dordrecht: Springer, 2009:181.

4. Binary numeration had already been suggested in 1689 by Leibniz, in his article *Explication de l'Arithmétique Binaire*, inspired by the Chinese I-Ching.

stand how the evolution of abstract thought (logical, mathematical, algorithmic, philosophical) has always been linked to technical developments.

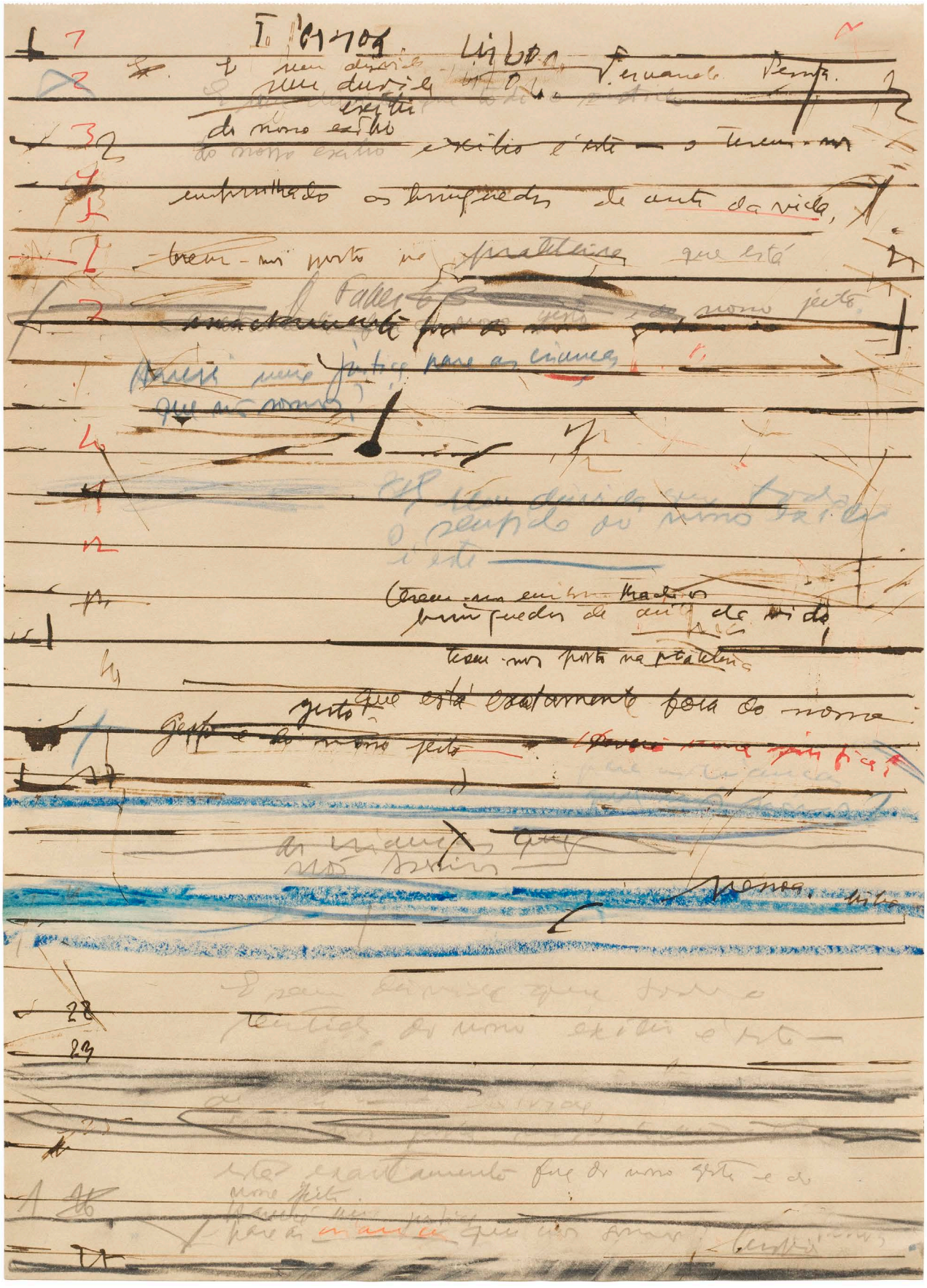
In the industrial era, the first complex algorithm to be mechanised was de Prony's method of differences, which Charles Babbage incorporated into his *Difference Engine*, with the idea of automating the calculation of logarithmic tables (at a time when sea routes were being used for colonial expansion and the tables were, among other things, used for determining longitude in navigation). Babbage's 'calculating engines' were never built because it was difficult to implement a decimal system in a simple and robust mechanical apparatus, but they are regarded as the first point of convergence between mathematical algorithms and industrial automation.

Subsequently, in the true Information Age, calculation algorithms were not mechanised but electrified thanks to the binary system: that is to say they were implemented in electrical and electronic circuits for faster results. To be precise, binary numbers were only adopted after 1940, in the wake of the magisterial thesis by the young Claude Shannon, 'A Symbolic Analysis of Relay and Switching Circuits', which was presented at the Massachusetts Institute of Technology. In this thesis, Shannon proposed using the binary properties of electrical switches to represent propositional logic and perform Boolean logical functions (denoted as AND, OR and NOT).[4] Unlike decimals in a moving mechanism, binary numbers are extremely easy to implement in an electronic circuit: the values 0 and 1 can easily be represented by the 'off' or 'on' state of an electrical circuit, for instance. With the introduction of the programmable electric computer, a complex change occurred: the numbers themselves became instructions. The so-called 'information revolution' concerned not only the use of binary numbers (binary digits or '*bits*') to codify human language and analogical content (by means of *digitisation*), but also the speeding up of mechanical calculation through binary logic (or *Boolean logic*). Computer languages and programmes gradually made it possible to represent complex algorithms, which ultimately always consist of the simple binary instructions of the 'machine code'.

In the twentieth century, binary code, Von Neumann architecture, and the implementation of 'logic gates' in efficient microchips, made it possible to build fast computers and to formalise huge and very complex informational algorithms. For the first time in history, sequences of numbers were used to represent not only quantities but *instructions*, that is to say algorithms (which had themselves originally been expressions of numerical algorithms).

"In seventeenth century Europe, philosophers like Pascal and Leibniz were already designing manual calculators for performing the four basic operations, using the decimal system."

Untitled, 1980



"The so-called 'information revolution' concerned not only the use of binary numbers (binary digits or '*bits*') to codify human language and analogical content (by means of *digitisation*), but also the speeding up of mechanical calculation through binary logic (or *Boolean logic*)."

Contrary to the general view that makes a clear distinction between *hardware* and *software*, digital calculation is in reality the implementation, in the same medium of information and instruction, of binary numbers and Boolean logic, the first being a complementary form of the second. With digital calculation, the numbering algorithm (*binary numbers*) and the calculating algorithm (*binary logic*) have for the first time become one and the same thing.

CONCLUSION

The history of numbers and algorithms cannot cease at a purely technical level, as it would risk remaining simply 'culturalist'. To summarise the stages of this very brief essay, we may posit that it was the commercial acceleration of the late medieval period, the industrial revolution, and the growth of the information society that formalised the algorithm as it is known today. Just as the medieval term *algorismus* marked the passage from the additive Roman system to the positional Indo-Arabic system (both decimal), the more recent term *algorithm* marks the passage from the decimal positional system to the binary system. Indian numbers and algorithms for manual calculation were adopted because they could speed up book-keeping and therefore commercial transactions, and binary numbers were adopted because they could be implemented in electrical circuits and microchips, thus enabling calculations to reach superhuman speeds. While the first transition was tied to commercial transactions, the second is linked to industrial capitalism and its drive towards the automation of manual and intellectual labour.

TRANSLATED BY EMMA MANDLEY / KENNIS TRANSLATIONS