Computing with concepts, computing with numbers

Uckelman, S.L.

Published in:
Programs, proofs, processes

Publication date:
2010

Document Version
Publisher's PDF, also known as Version of record

Citation for published version (APA):
Uckelman, S. L. (2010). Computing with concepts, computing with numbers: Llull, Leibniz, & Boole. In F. Ferreira, E. Mayordomo, & L. M. Gomes (Eds.), *Programs, proofs, processes* (pp. 427-437). Unknown Publisher.

Computing with Concepts, Computing with Numbers: Llull, Leibniz, and Boole

Sara L. Uckelman[⋆]

Institute for Logic, Language, and Computation
S.L.Uckelman@uva.nl

Abstract. We consider two ways to understand "reasoning as computation", one which focuses on the computation of concept symbols and the other on the computation of number symbols. We illustrate these two ways with Llull's *Ars Combinatoria* and Leibniz's attempts to arithmetize language, respectively. We then argue that Boole's development of an algebra of reasoning was in a large part successful due to its ability to marry the two types of computation that are exemplified in Llull's and Leibniz's works.

1 Introduction

The *Oxford English Dictionary* defines 'computation' as

> **1. a.** The action or process of computing, reckoning, or counting; arithmetical or mathematical calculation; an instance of this [1].

As with many dictionary definitions, this gloss is rather vague. There are (at least) two interesting ways in which it may made precise: one focusing on the words 'action, process' and the other on the words 'calculation, reckoning, counting, arithmetical, mathematical'. The first group of words suggests a mechanistic view of computation, whereby computation is the result of a machine running some algorithm or set of instructions. The second group of words, in that they all refer more or less directly to numbers, can be seen as a specification of the first group, by specifying that the actions or processes (mechanisms) involved are numeric or arithmetic. It should be clear that numeric processes and actions are not the only way to do computation; one non-numeric computational process or mechanism involves computation with concepts directly, and not via numerical representation. Thus, the two views of computation that will be considered in this paper are those indicated in the title: computation with concepts and computation with numbers.[1] These two branches or strands of computation are

[⋆] The author was funded by the project "Dialogical Foundations of Semantics" (DiFoS) in the ESF EuroCoRes programme LogICCC (LogICCC-FP004; DN 231-80-002; CN 2008/08314/GW).

[1] Strictly speaking, we should speak of "computation with concept-*symbols*" and "computation with number-*symbols* (or numerals)", since we are not directly operating on the concepts and numbers themselves. However, we will use the less precise formulation throughout the rest of the paper and trust the reader not to misunderstand our intent.

not intended to be exclusive, but rather, as we will see when we look at Boole, as complimentary.

In ordinary use, "computation" has some connotation of mindlessness. Many ordinary users of modern computers would subscribe to this view: computers are something of a black box where the user provides input and receives an output through a computational process which is often invisible, and even when visible, can often be wholly impenetrable (for example, watching LaTeX code compile). As we'll see, both Llull's *Ars Combinatoria* and Leibniz's attempts to arithmetize language have this mindlessness component: In so far as these systems are algorithmic, they move the burden of the actual reasoning from the user to the system. This property is illustrated in Boole's algebras by their level of abstraction.

In this paper we survey two different but connected ways that we can understand "reasoning as computation", and illustrate these ways by looking at the works of three figures in the history of logic, Ramon Llull, Gottfried Wilhelm Leibniz, and George Boole. Despite the well-documented inspiration that Leibniz found in Llull's works, Llull's views of the computational side of reasoning contrast with Leibniz's and Boole's development of an algebra of reasoning can be seen as a conceptual synthesis of the two. I argue that the success of Boole's innovation was due in large part to its ability to marry the two types of computation that are exemplified in Llull's and Leibniz's works: computation as rule-based manipulation of concepts and computation as arithmetic calculation.

The plan of the paper is as follows: To illustrate the conceptual view of computation, in the next section we look at a particular aspect of Ramon Llull's system of reasoning developed in the 13th century. Llull, who can rightfully be called a visionary for his ideas concerning the computational side of reasoning, inspired Leibniz in his development of a *calculus universalis* in the 17th century. One of Leibniz's goals along the way to the *calculus universalis* was the arithmetization of language, which serves as our example in Sec. 3 of the numeric or arithmetic view of computation. We then argue in Sec. 4 that these two strands are exemplified together in Boole's development of the algebra of reason, and it is their complementarity that was at least a partial cause of its success.

2 Llull and the Computation of Concepts

Ramon Llull (Catalan; *Raymundus Lullus* or *Lullius*, Latin; *Rámon Lull*, Spanish; *Raymond Lull* or *Lully*, English) was born in 1232 or early 1233 in Palma, the capital of Majorca, to a family that was probably of noble status. In his early years, he served as a courtier in the court of James I and James II of Majorca, which involved extensive travel through Aragon, Catalonia, and Valencia. In 1263 he experienced a religious conversion, and thereafter turned his attentions from secular pursuits such as troubadour poetry lyrics, to theological and philosophical topics. In the 1280s he conceived of a goal of developing a system of argumentation or demonstration which could be used to show the Jew and the Muslim the error of their ways, and the correctness of Christian theology. In addition to knowing Catalan, his native language, he was also conversant in

	Fig. A	Fig. T	Questions & Rules	Subjects	Virtues	Vices
B	goodness	difference	whether?	God	justice	avarice
C	greatness	concordance	what?	angel	prudence	gluttony
D	eternity	contrariety	of what?	heaven	fortitude	pride
E	power	beginning	why?	man	temperance	pride
F	wisdom	middle	how much?	imaginative	faith	accidie
G	will	end	of what kind?	sensitive	hope	envy
H	virtue	majority	when?	vegetative	charity	ire
I	truth	equality	where?	elementative	patience	lying
K	glory	minority	how? and with what?	instrumentative	pity	inconstancy

Fig. 1. The Alphabet of the *Ars brevis* [2, p. 581]

Latin and Arabic, the academic languages of his time. Part of implementing this goal involved missionary travel throughout the Mediterranean. He died during one of these journeys, either in Tunis or on a ship sailing from Tunis back home, sometime between December 1315 and March 1316.[2]

Llull, with his adventurous life, mysticism, and connections in high places, is an exciting figure for computer scientists to read and hear about. Given his adventures (cf. the citations in fn. 2), it's no wonder that computer scientists would want to claim such a celebrity as one of their own. Why they should do so is nicely argued by Sales in [4], who points out that in Llull's work can be found inklings of and first steps towards a number of concepts fundamental in computer science, such as the ideas of a calculus, an 'alphabet of thought', a method, a graph, of logical analysis, heuristics and deduction, generative systems, tableaux, conceptual nets, and diagrams to represent concepts and relations between concepts.[3]

Our interest here, in the context of "reasoning as computation", are just two out of Llull's numerous (263 according to [2, p. 53]) works, the *Ars demonstrativa* (Demonstrative Art, c. 1283–89, hereafter referred to as AD) and the *Ars brevis* (Short Art, 1308, hereafter referred to as AB), the "single most influential work", which builds on and simplifies AD. These works together constitute the 'Art'. In the Art, Llull presents to the reader a mechanism for abstract reasoning with a restricted range of application. The foundation of this mechanism is an alphabet, a system of constants, representing different concepts. The alphabet of AD is two-tiered, with 16 symbols representing basic concepts and then 7 symbols representing what we might call meta-concepts. In AB, the alphabet of AD is simplified to include only 9 symbols, and these symbols have different meanings depending on their usage. Fig. 1 gives the interpretation of the alphabet of AB in different contexts. The full combinatoric power of the Art comes to the fore in AB, where it "became a method for 'finding' all the possible propositions and syllogisms on any given subject and for verifying their truth or falsehood" [2, p. 575]. The allowed combinations of the constant symbols in the alphabet are

[2] This is a very compressed biography of Llull; for a more detailed history full of exciting details, see [2, vol. 1, pp. 3–52], which includes extensive excerpts from Llull's autobiography *Via coaetanea*, and [3, ch. 1].

[3] See [5] for further discussion of Llull's status as a 'computer scientist'.

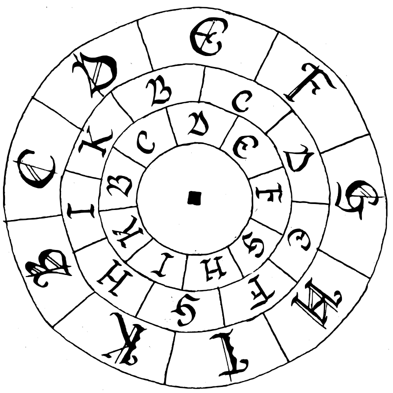

Fig. 2. The Fourth Figure

illustrated by various tables and diagrams[4] of which the most interesting, from the mechanist view of computation, is Figure 4 of the *Ars brevis*, which "has three circles, the outermost of which is fixed and the two inside ones of which are mobile" [2, p. 587]. Figure 4, redrawn from Plate XVIII, is given in Fig. 2. By rotating the moving circles in various ways, one can extract valid syllogisms, where the term on the middle circle is the middle term relating the two extremes, which are located on the outer and inner circles.

The physical nature of the concentric, movable circles of the fourth figure allows us to view this part of Llull's combinatorial system as a crude mechanism for computing new concepts (the output) on the basis of a given set of concepts (the input). As Welch says, "The fourth figure was thus a primitive logical machine" [6, p. 6], and Gardner calls it a "mechanical method" [3, p. 9]. Its primitiveness comes from both the rude nature of its construction, and also the fact that it essentially only handles intersection [7, p. 12]. But the primitiveness of it should not detract from its novelty: It is, so far as is known, the first attempt to provide a mechanistic and 'mindless' (in the sense of the word discussed above) *physical* method of reasoning.[5]

Llull's computational system is based purely on concepts; there is no arithmetic involved.[6] It is for this reason that we have selected Llull as an example of the purely non-arithmetic conception of reasoning as computation. However,

[4] The diagrams of the first, second, third, and fourth figures of the *Ars brevis* as found in the Escorial MS are reproduced in [2] between pages 582 and 583; a selection of figures from the Venice MS of AD, along with some interpretational tables by Bonner, are given in the same source between pages 318 and 320.

[5] We recognize that the introduction given here is nowhere near adequate. For further discussion of Llull's system, see [2] and [8].

[6] One should not take this criticism unfairly: Without the notions of the 'intension' and 'extension' of a concept (ideas not developed until the 17th century), it is by no means clear how one would associate numbers with concepts in any useful fashion that would allow the numeric type of computation.

this lack of mathematical foundation is often cited as one of the failures of his system, which is cumbersome and not easily extended (even though, as Zweig notes, "Llull believed his Art could be applied to all fields of knowledge and was therefore a truly universalist system" [9, p. 22]). Llull's mathematical naïvity was one factor which motivated Leibniz's search for a more rigorous and numerically-based system of computation [6, p. 2]. We turn to discuss this in the next section.

Before concluding this section, we note in passing that if we understand "computation" in the broad sense of "(algorithmic) process", then Llull was by no means the first to attempt a mechanistic view of reasoning. John of Salisbury, writing in the middle of the 12th century, tells us that his student, William of Soissons,

> invented a device to revolutionize the old logic by constructing unacceptable conclusions and demolishing the authoritative opinions of the ancients [10, Bk. II, ch. 10, p. 98].[7,8]

According to Kneale and Kneale, some people have thought that William's "machine" was a physical construction, akin to Jevon's logical machine [13, p. 201],[9] but both the Kneales and Martin have argued that "machine" should be understood here in a metaphorical sense, and that it was likely that what William had in mind was a method of argument-construction which, given a contradiction or an impossible statement, would return any other statement [14, p. 565]. Whether William's machine was a concrete object, or merely a procedure for a reasoner to follow, it is an interesting example of a computational method where the user is no longer necessarily the reasoner; rather, it is the "machine" itself which is doing the reasoning.[10]

3 Leibniz and the Computation of Numbers

Leibniz discovered Llull's works at an early age; he discusses Llull in his *Dissertatio de arte combinatoria* (Dissertation on the combinatorial art, 1666), written at the age of 19 [3, p. 3]. As Bonner notes, "the relational nature of Llull's system is fundamental to his idea of an Ars combinatoria" [5, p. 4]. Given Llull's emphasis on binary and ternary relations and his combinatoric approach to reasoning,

[7] *Interim Willelmum Suessionensem, qui ad expugnandam, ut aiunt sui, logice uetustatem et consequentias inopinabilies construendas et antiquorum sententias diruendas machinam postmodum fecit* [11, Bk. II, ch. 10, p. 81].

[8] Adamson [12, p. 27] translates John of Salisbury's *machinam* as "method", and the Kneales translate it as "engine" [13, p. 201].

[9] The Kneales do not say who these "some people" are, and I have been unable to find this out myself.

[10] All this talk of machines reasoning sounds anachronistic and, from the point of view of the 12th century, futuristic. Interestingly, from the point of view of contemporary artificial intelligence, while Leibniz believed that it was possible to mechanize (in the physical sense of the word) these algorithmic processes, "he never thought that we might invent a machine which could invent machines" [15, p. 110].

it is no surprise that he was such a fascinating figure to a young Gottfried Wilhelm Leibniz [5,6,16,17, p. 657]. Llull and Leibniz shared much in their guiding philosophy and goals. They both shared the desire to provide a system within which all theological controversies could be definitively solved. (Though they differed in the application of this system: Llull wanted to use it to show the Jew and the Muslims their errors, and the truth of the Christian way, whereas Leibniz intended his to help resolve the splintering of the Catholic church that had started in the previous century.) Both Llull and Leibniz also shared a belief in what Welch calls "conceptual atomism, a belief that the majority of concepts are compounds constructed from a relatively small number of primitives" [6, p. 2] (cf. [4, p. 16] and [17, p. 20]). These primitives could be combined and related in different ways which would allow the reasoner to generate complex information. Llull and Leibniz also both developed combinatorial systems which exploited physical systems of moving wheels [18, p. 274].[11]

However, Leibniz differed from Llull in two ways: As we noted in the previous section, he criticized Llull's mathematical naïvity and wished to ground his systems arithmetically, and he had access to the recently-developed notions of the intension (or comprehension) and extension of a concept. If we wish to speak anachronistically, we could say that, in so far as Llull's concepts could be represented in a physical medium and manually manipulated, he identified concepts with their extensions. Leibniz, on the other hand, discussed both intensional and extensional interpretations of concepts in his arithmetization of language, in the end preferring the intensional approach [20,21]. We offer Leibniz's attempts to arithmetize syllogistic as an illustration of the numeric interpretation of reasoning as computation. Leibniz made several different attempts to arithmetize Aristotelian syllogistic in his pursuit of developing a universal language, of which only the final attempt was successful. Given his belief in logical atomism, it was a natural step for him to associate primitive concepts with their *numeri characteristici*. If this is done properly, then more complex concepts could be reduced to their constituent parts merely by knowing both the rules for the combination of primitives and the mapping associating numbers with primitives, and further, syllogistic statements, which assert relations between complex concepts, could be represented by numeric relations between different numbers.

We consider two of the arithmetizations of syllogistic that Leibniz developed. Recall that a syllogism is a set of three categorical proposition, where a categorical proposition is one of the form "All S are P" or "Some S are P" (or their negations, "Some S are not P" and "No S are P", respectively, but since their

[11] Hence it should be clear in the following that we are not trying to argue that Leibniz did not have a mechanistic or purely concept-based view of reasoning; by no means is that the case, as is amply illustrated by his combinatorial theory, e.g., as presented in *Dissertatio de Arte Combinatoria*, which is much closer to the Llullian-style conceptual computation, relying less heavily on numeric support. This system is extensively discussed in [18, ch. 5] and [19, ch. 3]. Since in this paper we are interested in his developments which illustrate the approach of computing with numbers, we do not discuss his developments which fall under the approach of computing with concepts further here.

truth conditions fall straightforwardly out of the truth conditions for the affirmative claims, we will not consider them), where S and P are variables for terms representing primitive or complex concepts. Leibniz focused on the syllogism because he believed that all propositions could be reduced to subject-predicate ones, meaning that all reasoning could be simulated with syllogistic reasoning [19, p. 13]. In the first arithmetization attempt, each primitive concept is associated with a single positive integer; the characteristic number of a complex concept is the multiplication of all the primitive concepts it contains (for example, if 2 is 'animal' and 3 is 'rational', then 6 is 'rational animal'='man').[12] Then, a universal affirmative proposition is true iff the characteristic number of the subject term is divisible by the characteristic number of the predicate term [22, p. 42]. These truth conditions are acceptable if one adopts the constraint that no factor can appear in the characteristic number of a term more than once (cf. [21, p. 3]); that is, he recognized the idempotence of properties. How to treat particular affirmative propositions is less clear; Leibniz first offers the rule that such a proposition is true iff the characteristic number of the predicate is divisible by the characteristic number of the subject, or vice versa [22, p. 43]. However, this has the unfortunate consequence that a particular affirmative proposition "Some S is P" implies either that every S is P or that every P is S, which is not generally true. In a later manuscript, Leibniz revised the truth conditions for particular affirmative statements so that they are true whenever the characteristic number of the subject term is multiplied by another integer, it is then divisible by the characteristic number of the predicate term [22, pp. 58,69]. However, taken at face-value, this rule is also problematic; there is always some integer n such that sn is divisible by p, namely p itself. Thus, this rule implies that "Some S is P" is always true, which is unacceptable.

The second attempt at arithmetization [22, pp. 77–82] that we consider is more sophisticated. We follow the presentation of Marshall:

> Each term is assigned an ordered sequence of two relatively prime numbers, the first positive, the second negative. If each number assigned the predicate term divides the corresponding number of the subject term, the proposition is of the form 'All a is b'. The negation of this form. . . will be given if one of the two conditions is not met. . . If two of the non-corresponding numbers (i.e., the positive assigned one term and the negative assigned the other) have a common divisor, the proposition is of the type 'No a is b. If this is not the case. . . , the proposition is of the form 'Some a is b' [23, pp. 238–39].

This system has a number of points in its favor: It solves the problem of the triviality of the affirmative particular sentences, and it validates all the Aristotelian laws of conversion and valid syllogisms, and the square of opposition. But it comes with its own problems. Consider the following assignment of characteristic numbers to concepts: Let 'pious man' be assigned $\langle 2 \times 5, -3 \rangle$; 'fortunate

[12] The association of primitives with numbers which can be uniquely factored out of complex combinations of the primitives should strike the reader as reminiscent of Gödel-numbering.

man' be assigned $\langle 2^3, -11\rangle$, and 'happy man' be assigned $\langle 5, -1\rangle$; and the following syllogism:

> All pious men are happy.
> Some pious man is not fortunate.
> Some fortunate man is not happy.

On this assignment, the syllogism is verified (we leave the determination of this to the reader: it is straightforward). However, it should be immediately clear that this is not a valid argument; if we instead use the assignment of $\langle 2^4 \times 7, -3^3 \times 5\rangle$, $\langle 2^2, -3^9\rangle$, and $\langle 2, -3^3\rangle$, for 'pious', 'fortunate', and 'happy' respectively, then the syllogism is not validated. Leibniz concluded that there was an error in his arithmetization, since, while it validates all of the valid syllogisms, it failed to invalidate the invalid ones ([23, p. 240], [22, p. 334]). Note that if he had taken as a rule that "If invalidating instantiations can be produced, the mood is invalid; it is valid if they cannot" [23, p. 241], then Leibniz's system would have been a success (this is the argument of Marshall's article). Not recognizing this, Leibniz expressed dissatisfaction with the system presented above and ultimately gave up his attempt at arithematizing language. Leibniz's error, in his attempts, was in trying to identify the atomistic primitive concepts too closely with numbers, so that he could assign numeric properties to them. In the context of the present paper, we can say that in this endeavor, he went too far on the computation as (arithmetic) calculation interpretation.

4 Boole and the Synthesis

In Llull, we saw an example of a purely mechanistic, non-numeric system of computation, based on concepts. With Leibniz's attempts to arithmetize the syllogistic, we had an example of computation from the other end of the spectrum. Both of these systems have their shortcomings, but both have points in their favor. It was George Boole who took the best of both systems and created a reasoning-system that combines both types of computation, a system which continues today to be widely used and extremely fruitful: Boole's abstract algebras, which lead to the development of Boolean algebras (cf. [29]).

Conceptually, Boole's achievements rest heavily on Leibniz's [22, ch. viii]. As the Kneales note:

> Leibniz realized already in the seventeenth century that there is some resemblance between disjunction and conjunction of concepts on the one hand and addition and multiplication of numbers on the other, but he did not find it easy to formulate the resemblance precisely and then to use it as the basis of a calculus of logic. It was this that George Boole (1815–64) achieved in his *Mathematical Analysis of Logic* [13, p. 404].

Leibniz made "an attempt to improve the presentation of logic ... by the use of algebraic symbolism" [24, p. 158], though he was not as successful in doing as he had hoped [23, p. 241]. However, Boole was not familiar with Leibniz's work until *after* both *The Mathematical Analysis of Logic* and *Laws of Thought* were published [25, § 5].[13] (My thanks to one of the anonymous referees for this reference.) Kneale argues that the immediately effective sources for Boole were Gregory's "On the Real Nature of Symbolical Algebra" and De Morgan's four papers on "The Foundation of Algebra" [24, p. 160] and that:

> From these sources it was possible to collect two important discoveries: (i) that there could be an algebra of entities which were not numbers in any ordinary sense, and (ii) that the laws which hold for types of numbers up to and including complex numbers need not all be retained together in an algebraic system not applicable to such numbers [24, p. 160].

The importance of the *calculus of operations* in the development of English mathematics in the 19th century is discussed in great detail in [26]. Boole's crucial discovery was that "there could be an algebra of entities which were not numbers in any ordinary sense" and that the 'numeric' laws governing these entities need not be arithmetic [13, p. 405]. Boole's elements of study were class-designating concepts; that is, unlike Leibniz, he focused on the extension of concepts, rather than their intensions [27, pp. 4–5], [28, p. 29]. In this way, Boole's algebras can be seen as synthesizing the insights of both Llull, who treated concepts in an extensional fashion (speaking anachronistically) but without exploiting any arithmetic tools, and Leibniz, who saw the utility of associating arithmetic properties with intensions, but went too far the other direction in trying to map arithmetic properties directly onto the properties of intension.

The only question left for us to address in the remaining space is the extent to which Boole's algebras, viewed as a method of computation, have the 'mindlessness' property discussed in the opening section. It is certainly the case that one can do these computations by rote, without having any idea of the meaning (interpretation) of the symbols being manipulated.[14] On the other hand, Boole himself allowed the mindless application of symbolic manipulation only if this was an intermediary step eventually followed by an active final step of reasoning involving expanding the symbols to their meanings [29, pp. 173, 179]. He says:

> It is of most material consequence, whether those symbols are used with a full understanding of their meaning, with a perfect comprehension of that which renders their use lawful, and an ability to expand the abbreviated forms of reasoning which they induce, into their full syllogistic

[13] Even after Boole was introduced to Leibniz's work, it is not clear what the extent of his access was; during Boole's lifetime many of Leibniz's works languished unedited, and it was not until Couturat's edition in 1903 [22] that Leibniz's important works became generally available. In particular, it is not known whether Boole knew of the *Non Inelegans Specimen Demonstrandi in Abstractis* (in the Erdmann edition of 1840), which "was the only important piece of Leibniz on mathematical logic then generally available. . . [Leibniz's] most interesting papers lay still unread in the library at Hanover" [24, p. 150].

[14] This fact is recognized in the first sentence of the introduction to Boole's *Mathematical Analysis of Logic*: "[T]he validity of the processes of analysis does not depend upon the interpretation of the symbols which are employed, but solely upon the laws of their combination" [27, p. 3].

development; or whether they are mere unsuggestive characters, the use of which is suffered to rest upon authority [27, p. 10].

Thus, we end with a two-faced observation: The first successful attempt at developing a system of computational reasoning was done by someone who would only admit the mechanistic approach towards deduction as a preliminary step. Even if we can have computers do our reasoning, it is still up to us to interpret and apply the results.

References

1. OED: Computation, *n.* In: Oxford English Dictionary. September 2009 edn. OUP (2009), `http://dictionary.oed.com/cgi/entry/50045985` (January 5, 2010)
2. Llull, R., Bonner, A. (ed. and trans.): Selected Works of Ramon Llull, 1232–1316, 2 volumes. Princeton University Press, Princeton (1985)
3. Gardner, M.: Logic Machines and Diagrams. McGraw-Hill, New York (1958)
4. Sales, T.: Llull as computer scientist. In: Bertran, M., Rus, T. (eds.) AMAST-ARTS 1997, ARTS 1997, and AMAST-WS 1997. LNCS, vol. 1231, pp. 15–21. Springer, Heidelberg (1997)
5. Bonner, A.: What was Llull up to? In: Bertran, M., Rus, T. (eds.) AMAST-ARTS 1997, ARTS 1997, and AMAST-WS 1997. LNCS, vol. 1231, pp. 1–14. Springer, Heidelberg (1997)
6. Welch, J.R.: Llull, Leibniz, and the logic of discovery (Revised version of "Llull and Leibniz: The Logic of Discovery". Catalan Review 4(1-2), 75–83 (1990), `http://spain.slu.edu/faculty_&_staff/directory/files/llull.pdf`
7. Styazhkin, N.I.: History of Mathematical Logic from Leibniz to Peano. MIT Press, Cambridge (1969)
8. Bonner, A.: The Art and Logic of Ramon Llull: A User's Guide. Brill (2007)
9. Zweig, J.: Ars Combinatoria: Mythical systems, procedural art, and the computer. Art Journal, 20–29 (Fall 1997)
10. Salisbury, J., McGarry, D.D. (trans. & ed.): The Metalogicon of John of Salisbury. University of California Press, California (1955)
11. Salisbury, J., Webb, C. (ed.): Ioannis Saresberiensis episcopi Carnotensis Metalogicon. Libri IV. E Typographeo Clarendoniano (1929)
12. Adamson, J.W.: A Short History of Education. Cambridge University Press, Cambridge (1919)
13. Kneale, W., Kneale, M.: The Development of Logic. rev. edn., Clarendon (1984)
14. Martin, C.J.: William's machine. J. Phil 83(10), 564–572 (1986)
15. Schrecker, P.: Leibniz and the art of inventing algorisms. J. Hist. Ideas 8(1), 107–116 (1947)
16. Wilson, C.: Leibniz's Metaphysics. Princeton University Press, Princeton (1989)
17. Leibniz, G.W., Loemker, L.E. (eds. & trans.): Philosophical Papers and Letters, 2nd edn. D. Reidel (1969)
18. Maat, J.: Philosophical Languages in the Seventeenth Century: Dalgarno, Wilkins, Leibniz. Kluwer Academic, Dordrecht (2004)
19. Ishiguro, H.: Leibniz's Philosophy of Logic and Language, Duckworth (1972)
20. Rescher, N.: Leibniz's interpretation of his logical calculi. J. Sym. Log. 19(1), 1–13 (1954)

21. Sotirov, V.: Arithmetizations of syllogistic *à la* Leibniz. J. App. Non-Class. Log. 9(2-3), 387–405 (1999)
22. Couturat, L.: Opuscules at Fragments inédits de Leibniz. F. Alcan (1903)
23. Marshall Jr., D.: Łukasiewicz, Leibniz, and the arithmetization of the syllogism. Notre Dame J. Form. Log. 18, 235–242 (1977)
24. Kneale, W.: Boole and the revival of logic. Mind 57(226), 149–175 (1948)
25. Peckhaus, V.: Leibniz's influence on 19th century logic. In: Zalta, E.N. (ed.) Stanford Encyclopedia of Philosophy (Fall 2009 edn.), `http://plato.stanford.edu/archives/fall2009/entries/leibniz-logic-influence/`
26. Koppelman, E.: The calculus of operations and the rise of abstract algebra. Archiv. Hist. Exact Sci. 8, 155–241 (1971)
27. Boole, G.: The Mathematical Analysis of Logic. MacMillan, Basingstoke (1847)
28. Boole, G.: Laws of Thought. Dover Publications, Inc., New York (1854)
29. Hailperin, T.: Boole's algebra isn't boolean algebra. Math. Mag. 54(4), 173–184 (1981)