CHAPTER XYZ

Leibniz's Characteristica Universalis And Calculus Ratiocinator Today

Bruno Woltzenlogel Paleo

Leibniz's dream of a universal language and a calculus for reasoning is well summarized in his own famous words (translated by Bertrand Russell): "If controversies were to arise, there would be no more need of disputation between two philosophers than between two accountants. For it would suffice to take their pencils in their hands, to sit down with their slates and say to each other (with a friend as witness, if they liked): Let us calculate." (Russell, 1958)[p. 170]

This dream has inspired several generations of logicians, mathematicians, computer scientists and engineers to build machines capable of understanding increasingly sophisticated languages. But what inspired Leibniz himself? In his *On the General Characteristic* (1679)[1], he admits that "learned men have long since thought of some kind of language or universal characteristic by which all concepts and things can be put into beautiful order". He cites the Pythagorean belief that "the deepest mysteries lie concealed in numbers" and talks about an Adamic language (i.e. a language spoken by God to Adam and Eve in the Garden of Eden, a language of Nature). His own never fully completed project of *Catholic Demonstrations* was a major motivation for the pursuit of a logic on which the demonstrations would be based. Antognazza (Antognazza, 2009)[p. 93] suggests that Leibniz was influenced by Thomas Hobbes's *Computatio sive Logica* (1655) and mentions several British and German authors who tried to create universal languages before Leibniz.

[1] Leibniz's works cited here can be found in (Leibniz, Philosophical Papers and Letters, 1956), (Leibniz, Sämtliche Schriften und Briefe, 1999) or (Leibniz, Sämtliche Schriften und Briefe, 2006).

Under the assumption that the creation and understanding of language is an innate human ability, it is likely that the general idea of developing a universal and absolutely precise language will necessarily occur throughout history to many people, and perhaps even to almost anyone who cares deeply enough about the limitations and imprecisions of ordinary language.

Leibniz himself tries to distinguish his work from those of his predecessors by saying that "yet no one has attempted a language or characteristic which includes at once both the arts of discovery and of judgment, that is, one whose signs or characters serve the same purpose that arithmetical signs serve for numbers, and algebraic signs for quantities taken abstractly" (*On the General Characteristic*, 1679). Leibniz's uses letters to denote concepts and special symbols to denote conjunction and equality of concepts (Lenzen, Das System der Leibniz'schen Logik, 1990).

The analogy between concepts and numbers is even more clear in his *Two Studies in the Logical Calculus* (1679), when he writes that "since man is a rational animal, if the number of animal is a, for instance, 2, and the number of rational is r, for instance, 3, the number of man, or h, will be 2 x 3 or 6". Leibniz's previous work (completed in 1673) on a prototype of a calculating machine more sophisticated than Pascal's (capable not only of addition and subtraction, but also multiplication, division and extraction of roots) (Antognazza, 2009)[p. 145] probably played a fundamental role in the formulation of the analogy. Leibniz was trying to encode the problem of reasoning about concepts into the simpler problem of reasoning about numbers, for which machines were being developed at his time. This pioneering approach is an early precursor of today's encodings of all sorts of problems down to binary arithmetic, which allows them to be solved by our ubiquitous modern computers.

Today, three centuries after Leibniz's death, it is only natural to wonder how much of Leibniz's dream has been fulfilled so far. To partially answer this question, this chapter gives a brief overview of current automated reasoning technology and its logical foundations, relating them to Leibniz's desiderata and to his own algebra of concepts.

Leibniz's Algebra of Concepts and the Logics of Today

Leibniz's pursuit of a characteristica universalis led him to invent an Algebra of Concepts[2]. In today's terminology, the language of this algebra is a first-order language where terms are concepts. It has primitive function symbols for concept negation and concept conjunction (which should not be confused with logical negation and conjunction of propositions), and a primitive relation symbol for equality of concepts. From these primitive functions and relations, Leibniz defined the relation of concept containment and the predicates of possibility and necessity. The notions of possibility and necessity apply to concepts; therefore, they should not be confused with the notions of possibility and necessity of propositions that are common in modern modal logics.

Although concept variables occurred in terms, formulas did not have explicit quantifiers. Instead, concept variables were implicitly assumed to be existentially or universally quantified depending on the context. This potential source of ambiguity[3] is worth noting, considering that Leibniz aimed at a perfectly precise language.

With hindsight it is clear that, being a first-order language, Leibniz's language was not expressive enough to be the universal language that he desired. Due to the Löwenheim-Skolem theorems, first-order logics lack categoricity. Furthermore, prominent axioms, such as the induction axiom and the axiom of choice, require higher-order quantification over predicates and functions. In fact, Leibniz's language has no place for uninterpreted function symbols at all, which is inconvenient and somewhat surprising, considering Leibniz's contributions to differential and integral calculus, where functions are first-class objects of discourse.

---

[2] A detailed presentation of Leibniz's Algebra of Concepts can be found in Lenzen's book (Lenzen, Das System der Leibniz'schen Logik, 1990).

[3] The implicitness of quantification in Leibniz's algebra is possibly a convention inherited from Mathematics. Still today we write equations like, for instance, $x^2 = 9$, instead of $\exists x.\, x^2 = 9$.

Today, the best candidates to be considered universal formal languages are the higher-order logics based on type theories, which form the basis of proof assistants such as Coq (Paulin-Mohring, 2015) and Isabelle (Wenzel, 2015). The universality of these logics, from both theoretical and practical points of view, is evidenced by their ability to embed/encode other logics (and even simulate Turing machines) and by their application in many different domains, including Mathematics, Software and Hardware Verification, and even Metaphysics.

From No Universal Language to Many Universal Languages

While the problem in Leibniz's time was the lack of a universal language, the problem today is that there are perhaps too many universal languages (and even more less universal ones). Although on the surface many of today's higher-order logics may seem similar to each other, they tend to differ drastically in their foundations. For instance, whereas Coq is based on an intuitionistic logic where formulas are types of dependently typed lambda calculus and principles like extensionality do not hold, Isabelle/HOL is based on classical logic where formulas are terms of a polymorphic typed lambda calculus and extensionality principles do hold.

From a computational point of view, most of such candidate logics, despite their foundational disagreements, can be considered equivalent to each other at least in the sense that theorem proving systems based on them are Turing-complete and any Turing machine can be simulated, at least in principle, by a theorem proving problem within these logics. However, from a social point of view, our inability to agree on one universal logic is a sign that, maybe, none of our existing languages is universal enough yet. There is no formal language that is universally used by every logician yet.

An answer to the question of whether Leibniz's dream of a universal language has already been achieved depends, therefore, on our interpretation of "universal". For most interpretations, the answer is clearly positive, but the social aspect, which conveys a desire for uniqueness of the universal language, should not be

ignored, since it is a concern that Leibniz also had in mind. Because Language is inherently social, it is constantly evolving: we are always inventing new ways to express ourselves. This is visible in the evolution of systems like Coq and Isabelle. Although their minimalistic core is quite stable, libraries and user interface features that make the systems more user-friendly have been evolving rapidly, and new versions are often not compatible with older versions.

It is ironic that Leibniz envisioned a universal formal language as a solution to the problem of the plurality of (interpretations of) natural language(s), but now we have so many of such formal languages and we continue to face the problem of plurality, just in a more formal level. In fact, there are even regular workshops, such as the Workshop on Proof Exchange for Theorem Proving and the Workshop on User Interfaces for Theorem Provers (Benzmüller & Woltzenlogel Paleo, Proceedings, 2014), focused on topics that include standardization of formal languages and interoperation of reasoning systems.

Less Expressive Formal Languages for Full Automation

When Leibniz writes about his vision of a universal language and of a reasoning calculus, he seems excessively optimistic. He apparently believes that, as soon as we develop a sufficiently expressive formal language, all problems and controversies will be just a matter of mechanical calculation. The problem is that, as we now know, there is usually a trade-off between expressiveness and efficiency. The more expressive and universal a logical language is, the less efficient theorem provers for that language tend to be. This observation led to a proliferation of less expressive formal languages and corresponding reasoning tools in the 20th and 21st centuries, with the aim to tailor the languages and the tools to special problems.

For example, to mention just a few types of reasoning tools, today there are sat-solvers for propositional logic, SMT-solvers for fragments of first-order logic extended with theories, description logic reasoners for other fragments of first-order logic relevant for ontologies, planners for logic fragments relevant for planning

problems, automated theorem provers for pure first-order logic, for first-order logic with equality, and for first-order logic extended with linear arithmetic, and various modal provers for various modal logics.

For a long time, systems that aimed on a very expressive and universal formal language had to give up on automation and focus on interactive reasoning guided by the user, whereas system that aimed on efficient automation had to give up on expressiveness and universality. But more recently, a new trend has emerged, which bring new hope for Leibniz's vision. The idea is to integrate both kinds of systems: the interactive theorem prover, guided by the user, can delegate easy and tedious parts of the problem to automated tools and automatically reconstruct their solutions within their own trusted logic.

General User-Friendliness

Leibniz desires a universal language and a reasoning calculus that would be easy to use. In his *Letter to Johan Friedrich von Hannover* (1679), he writes that his goal is "une langue ou écriture nouvelle, qui se pourroit apprendre en une semaine ou deux" ("a new language or script, that could be learned in one week or two") by virtually anyone. Today, we are still quite far from this ambitious goal. Modern logic is usually only taught to computer science students, and even then a basic course on Logic usually takes a whole academic semester. Interactive proof assistants are known to have very steep learning curves and even a talented student would typically only know the basics of such systems after one or two weeks of self-study or of attendance of a summer school course.

In relation to user-friendliness, current technology is still far from what Leibniz envisioned. Nowadays, everyone and anyone can use calculators in their mobile phones, but only a comparatively much smaller group of specialized researchers uses reasoning tools in their laptops and desktop computers.

For instance, the famous formalizations of proofs of difficult theorems, such as Kepler's Conjecture and the 4-Colour Theorem

took years to complete and relied on the efforts of several experts in the field of automated and interactive theorem proving.

Leibniz believed that logic (in the form of a universal language) would render science popular and familiar: “elle rendroit ce grand secret [sciences] populaire, et familier” (*Letter to Johan Friedrich von Hannover*, 1679), but nowadays advanced logic is still only available to few.

Machine Language versus Human-Readable Language

Leibniz seems to assume that the results obtained by a person or a reasoning machine, operating according to the rules of a logical calculus, on statements written in a precise universal language would be easy to comprehend. But this is not necessarily the case, and it is certainly not yet the case.

Sometimes, a simple “yes” or “no” answer for whether a statement is a theorem suffices, but frequently it is also important to know why the statement is a theorem or not. The languages used by automated reasoning systems to output proofs or counter-models are often poorly documented and still much less well-developed and standardized than their input languages (Schulz & Sutcliffe, 2015). They are also typically machine-oriented, and not designed for human-readability. For example, when the higher-order prover Leo-II detected an inconsistency in the axioms of Gödel’s ontological argument, the proof it generated was so cryptic, that it took several months to read and comprehend it well enough to formulate an intuitive informal explanation for the inconsistency (Benzmüller & Woltzenlogel Paleo, The Inconsistency in Gödel's Ontological Argument: A Success Story for AI in Metaphysics, 2016).

There is a mismatch between human and machine abilities in relation to logical reasoning. The kinds of inference that humans do efficiently are different from the kinds of inference that machines do efficiently. The resolution rule, for instance, which underlies many current automated provers, is difficult for humans to read, partly because it relies on the unification. Whereas unification is comparatively easy for a machine, it can be very

time-consuming for humans, especially when dealing with very large terms. The substitutions used in unifications are often a crucial information for understanding a formal proof, and yet this information is often not explicitly given in the proofs output by automated provers.

The Size of Proofs

Another problem that jeopardizes the hope of being able to fully understand machine-generated proofs is their size. In many cases, especially when an exhaustive combinatorial search is necessary, the theorem proving capacity of computers has already vastly exceeded human capacity. Consequently, they generate proofs that are too large for humans to check. Currently, the largest known automatically generated proof, which solves the Pythagorean Triples problem, occupies 200 Terabytes.

Since it is practically impossible for a human to check such large proofs, the current trend is to implement computer programs that check proofs (Miller, 2015) (Heule & Biere, 2015) as well as programs that compress proofs (Boudou, Fellner, & Woltzenlogel Paleo, 2014) (Fontaine, Merz, & Woltzenlogel Paleo, 2011). However, although they definitely help in increasing the confidence in the correctness of the checked proofs, they are not (yet) able to extract an insightful explanation for why the theorem holds.

Discovery and Invention

Leibniz repeatedly mentions that his envisioned language and calculus would aid people not only to judge, but also to invent and discover. He is not completely clear, however, about what he means by inventing and discovering. We can interpret him in various ways.

The most likely interpretation is that he believed that a symbolic language for general reasoning will aid the discovery of new facts and invention of new concepts, in the same way that a symbolic language facilitated the advancement of mathematics. It is easier to think about abbreviated symbols in a language with a precise

semantics than to think about texts written in a natural language. It becomes easier to discover patterns and regularities, and invent new concepts to talk about them.

Another possible interpretation is that the universal language and the calculus would enable the automatic discovery of interesting new theorems through application of the inference rules to axioms and lemmas that are already known. Such an application of automated reasoning has not been thoroughly pursued yet. Today's automated reasoners could be characterized as "conjecture checkers". Their input is a set of axioms and a conjecture, and their output is an answer whether the given conjecture is entailed by the axioms (and possibly a proof of this fact) or not (possibly with a counter-model to show that it is not). Today's reasoning tools are not "theorem discoverers" that would generate a list of interesting theorems derivable from a given set of axioms. Therefore, under this interpretation of discovery, we are still far from achieving Leibniz's dream. The challenge in achieving this sort of automatic discovery lies in the subjectivity of what makes a theorem interesting. The blind and exhaustive application of inference rules generates many theorems and it seems hard to conceive an objective method to filter only the interesting ones. A less ambitious but related problem is the problem of discovering interesting lemmas that could compress and improve the proofs of already known theorems. This is essentially Hilbert's 24th problem, and there have already been attempts to solve it (Boudou, Fellner, & Woltzenlogel Paleo, 2014) (Dunchev, Leitsch, Libal, Weller, & Woltzenlogel Paleo, 2010) (Woltzenlogel Paleo, 2014).

Yet another interpretation of inventing through formal logic would be the automatic synthesis of solutions to problems declaratively specified in a logical language. Logic programming, constraint solving and answer set programming can be considered successful realizations of this idea in the restricted case where the goal is to synthesize existentially quantified first-order variables. The harder problem of synthesizing function variables is still open in practice and is currently a very active topic of research, although theoretical interest in this problem can be traced back at least to Gödel's Dialectica interpretation, which is a method for

synthesizing a function f^* that would be a satisfying instance for $\exists f.\forall x.P(x,f(x))$ given a proof of $\forall x.\exists y.P(x,y)$ with certain restricted characteristics.

Proofs as Programs: The Brouwer-Heyting-Kolmogorov Interpretation and the Curry-Howard Isomorphism

A central idea in Leibniz's dream is that logical reasoning is a form of (or can be reduced to) computation. The most common understanding of this idea, and probably the only one intended by Leibniz, is that claims can be written precisely as logical statements in a universal language, and whether a claim is valid can be checked mechanically, through a computer program that methodically searches for a proof of the logical statement.

In the 20th century, however, a second kind of connection between logic and computation was gradually discovered in the context of constructivism and intuitionism. It became clear that proofs themselves could be seen as programs, and the intuitionistic theorems that they prove are the types of these programs. This correspondence between proofs and programs and between formulas and types is known today as the Curry-Howard Isomorphism. It is particularly easy to see by comparing the inference rules of natural deduction for minimal logic and the typing rules for simply-typed lambda calculus. However, this is a general phenomenon that is related to the so-called Brower-Heyting-Kolmogorov interpretation and applies to intuitionistic logics in general, including highly sophisticated ones such as the calculus of inductive constructions underlying the Coq proof assistant. Extending these ideas to classical logics is currently an active topic of research.

These two kinds of connections between logic and computation are subtly but fundamentally different. In the first kind, we have a program that does proof search, whereas in the second kind, the proof itself is a program (and its execution corresponds to normalization of the proof). Furthermore, the second kind (proofs-as-programs) should not be confused with extraction of programs from proofs through Dialectica-style synthesis methods. In the former, the proof itself is a program, whereas in the second a

program is extracted from a proof of a logical statement of a particular form. These two programs do not coincide, although they are closely related.

From Leibniz's time to our days, our understanding of the connection between logic and computation has clearly become sharper: we see now that not only one but several connections can be established.

The Curious Challenge of Combining Logic and Arithmetic

Leibniz's success in inventing machines that could evaluate arithmetical expressions was an inspiration for him to dream of doing something similar with logical expressions instead. The idea that conjunctions in his Algebra of Concepts could be encoded as multiplication of numbers, is an early sign of what we have today. At the lowest level, all our high-level logical abstractions become 0s and 1s in a computer.

Considering that logical reasoning is ultimately encoded as binary arithmetical computations, it may seem ironic that one of the major challenges faced by automated reasoning today is the combination of logical and arithmetical reasoning. We have efficient reasoning systems for pure logic and powerful methods for doing all sorts of mathematical calculations, but systems that attempt to do both (e.g. SMT-solvers and ATPs modulo theories) are often limited to weak fragments of arithmetic (e.g. only linear arithmetic) and do not perform as efficiently as the corresponding systems for pure logic (e.g. sat-solvers and pure ATPs). This apparent paradox is easily resolved by noticing that the arithmetical evaluations performed by Leibniz's calculating machines and even the binary arithmetic of current computers are all ground (i.e. without variables), whereas the challenging problems that result from combining logic with arithmetic contain existentially and universally quantified variables within arithmetical expressions. Solving such problems requires solving systems of arithmetical (in)equations, and this is fundamentally different and more difficult than merely evaluating ground expressions.

Another practical open problem in the journey to extend automated reasoning from pure logic to logic modulo mathematical theories is induction. To prove that a property P holds for all natural numbers, it is typically necessary to prove this by induction. However, it is sometimes the case that the property needed to instantiate the induction axiom is not P itself, but another property Q stronger than P. Guessing the property Q is non-trivial, and exhaustively trying all possibilities is hopelessly inefficient.

Natural Language Translation

Leibniz's universal language ought not only to allow precise reasoning but also to be such that "with whose help different nations might communicate their thoughts and each read in his own language what another has written in his" (*On the General Characteristic*, 1679). In other words, the universal logical language could serve as an intermediary language when translating from one language to another.

Such a logical approach to natural language translation has been tried. However, although automatic translation is already widely available today (through web services such as Google Translate, Yandex Translate and Bing Translator), their algorithms are based primarily on statistics and machine learning instead of logic. They make use of the vast amount of data available in several languages in the web in order to calculate the most plausible translation of a natural language sentence without representing the sentence's meaning in logical form. Despite the success of the statistical approach, it is plausible that there is a limit to the quality of the results it produces on its own and that a combination of both approaches could lead to an improvement in quality.

Contrary to what may be implied by Leibniz's optimistic statement, representing a natural language sentence by a logical formula that correctly captures its meaning is not trivial. There are not only syntactic parsing challenges and ambiguities to be solved, but also semantic and pragmatic difficulties originating, for instance, from implicit presuppositions and implicatures that are

indirectly conveyed in sentences and discourses (Lebedeva, 2012). Moreover, whereas mainstream formal logical languages seem sufficient to easily represent simple natural sentences in the indicative mood, it is much less clear how to formally represent sentences in subjective, imperative and interrogative moods, for instance.

Domains of Application

Leibniz desired a "general characteristic able of achieving, in all fields of inquiry capable of certainty, what algebra does in mathematics" (*Letter to Biber* (1716) (Antognazza, 2009)[p. 528]). That's yet another sense of universality which, however, has not yet been fully accomplished. With the exception of occasional knowledge bases, logic-based expert systems and decision support systems, usually built with special-purpose languages and logics for domains varying from common sense reasoning to medicine and even highway management, most general-purpose automated reasoning tools have been applied primarily to mathematics and to computer science (e.g. formal methods for hardware and software engineering). Leibniz himself was particularly interested in physics and metaphysics, but modern uses of logic in these two disciplines are comparatively few and not yet mainstream in venues dedicated to the development of automated and interactive reasoning theory and technology. Nevertheless, works relating logic and physics can be found, for instance, in independent venues such as the series of workshops on Physics and Computation (Andréka, Madarasz, Németi, & Székely, 2010) and the series of conferences on Logic and Relativity (Madarász & Székely, 2015); and interest in metaphysical applications has started with the formalization of Anselm's and Gödel's ontological arguments (Oppenheimer & Zalta, 2011) (Benzmüller & Woltzenlogel Paleo, Automating Gödel's Ontological Proof of God's Existence with Higher-Order Theorem Provers, 2014), and evidence of its growth can be seen in the first World Congress of Logic and Religion. Other modern disciplines, such as economics, including social choice theory and game theory, might prove to be profitable research fields for logic applications as well.

Incomprehensible Artificial Intelligence

A strong interpretation of Leibniz's view encourages us to delegate decisions on controversial matters to machines. We expect the machine not only to tell us what to accept or do, but also to rationally explain us why. The machine's decisions and justifications are tacitly assumed to be comprehensible. Although this assumption might have been reasonable at Leibniz's time, it is questionable today.

In Leibniz's time, calculating machines were comparatively simple and the decision-making process envisioned by Leibniz would be based on well-known and accepted logical principles. But today's computers use sophisticated heuristic machine learning algorithms to interpolate and extrapolate from vast amounts of data. The internal representation of the automatically acquired knowledge is often not based on logic, and the decision is not reached through logical reasoning in a strict sense. For instance, AlphaGo, the computer program that beat the 9-dan professional human Go player Lee Sedol analyzed several real and simulated games to learn large deep neural networks for values and policies, and employed a monte carlo tree search method that uses the neural networks to compute the next move. Some of its moves surprised expert human observers, who could not immediately comprehend the reasons behind AlphaGo's choices in terms of their own knowledge of the game. This sort of decision-making technology is currently being deployed not only in harmless game playing programs but also in, for example, automatic trading algorithms and in self-driving cars. Such critical contexts of use have led to growing concerns about the safety and ethics of decisions taken by potentially incomprehensible artificial intelligence. These concerns call for the further development of logic-based solutions, such as (so called good old-fashioned) logic-based AI approaches, logic-based verification of the safety and ethical conformance of AI approaches based on machine learning or general AI approaches with some sort of self-introspective capacity to explain, justify and certify its own decisions.

It is ironic that the dream of computers taming our own incomprehensible controversies and irrationalities through rigorous logic might be just turning into a nightmare, as computers themselves become increasingly incomprehensible to us through the use of approximate, probabilistic and big-data-based algorithms. The quest for logic-based solutions to this nightmare might be a modern reincarnation of Leibniz's dream.

But machine learning is not only a challenge for logic. It is also a tool that can be useful to improve the efficiency of automated reasoning. Machine learning algorithms have been used to learn the best parameters of a theorem prover for every problem or to select the axioms and lemmas that seem promising to prove a given conjecture (Urban, Sutcliffe, Pudlák, & Vyskočil, 2008).

Universality as Logic's Weakness

The universality of logic is also its weakness. As usual, the more universal and general a method is, the less useful to specific problems it tends to be. To become sufficiently efficient, logic-based solutions often need to be tweaked and optimized to a point where they might not be recognized as logic anymore. A notable example is database technology. A database is essentially a collection of atomic axioms, and SQL is essentially a logical language to express conjectures. Logic is everywhere, but it is hard to see. It is often reinvented with different terminology and its popular and successful spin-offs may lose touch with their origins. Therefore, the importance of logic for applications is underestimated and, consequently, not adequately rewarded. Not surprisingly, despite being one of the oldest disciplines, discussed already by Aristotle, logic remains fragmented and logicians still survive scattered in philosophy, mathematics and computer science and engineering departments.

Conclusions

Although Leibniz's technical contributions to logic (i.e. his Algebra of Concepts) did not survive into mainstream modern logic, his dream of a characteristica universalis and a calculus ratiocinator have inspired several generations of logicians. The

dream is more alive than ever. It is estimated that the Vienna Summer of Logic (which united most major logic conferences) attracted 2000 logicians to the Austrian capital for a period of two weeks in 2014. And in October 2016, a Dagstuhl seminar on the "Universality of Proofs" will bring 38 experts to the famous German castle to discuss precisely the topics that interested Leibniz more than 300 years ago: universal logical languages and general automatic and interactive reasoning methods. If the topic is old, the technology is not. There is no doubt that we have seen an immense progress in the field of Logic in the last three centuries, especially since the dissemination of computers in recent decades. We can consider that Leibniz's dream has already been mostly fulfilled. However, as discussed here, there are still challenges ahead to satisfy many of his desiderata.

Leibniz remained faithful to his dream throughout his life. In his *Letter to Biber*, written in the year of his death (1716), he writes: "[…] if God gives me more strength thereafter, I will try to discharge myself of certain ideas I still have regarding the advancement of knowledge. For I see that it is possible to develop a general characteristic able of achieving, in all fields of inquiry capable of certainty, what algebra does in mathematics." (Antognazza, 2009)[p. 528]

Defending logic, he once wrote:

"Je croy donc que cette Methode revestue de la forme d'une langue ou characteristique est la plus importante chose que les hommes puissent jamais entreprendre pour l'avancement des sciences." (*Letter to Johan Friedrich von Hannover*, 1679)

"I believe that this Method, coated with the form of a language or characteristic, is the most important thing that men could ever undertake for the advancement of the sciences."

Bibliography

Andréka, H., Madarasz, J., Németi, I., & Székely, G. (2010). Axiomatization of Relativistic Physics in a Logical Framework. *3rd International Workshop on Physics and Computation.*

Antognazza, M. R. (2009). *Leibniz: An Intellectual Biography.* New York, NY, USA: Cambridge University Press.

Benzmüller, C., & Woltzenlogel Paleo, B. (2013). Automated Verification and Reconstruction of Gödel's Proof of God's Existence. *Journal of the Austrian Computer Society*, 4-6.

Benzmüller, C., & Woltzenlogel Paleo, B. (2013). Formalization, Mechanization and Automation of Gödel's Proof of God's Existence. *arXiv, 1308.4526*. arXiv.

Benzmüller, C., & Woltzenlogel Paleo, B. (2013). Gödel's God in Isabelle/HOL. *Archive of Formal Proofs*.

Benzmüller, C., & Woltzenlogel Paleo, B. (2013). Gödel's God on the Computer. *10th International Workshop on the Implementation of Logics*, (pp. 1-2).

Benzmüller, C., & Woltzenlogel Paleo, B. (2014). Automating Gödel's Ontological Proof of God's Existence with Higher-Order Theorem Provers. *European Conference on Artificial Intelligence - Frontiers in Artificial Intelligence and Applications. 263.* Prague: IOS Press.

Benzmüller, C., & Woltzenlogel Paleo, B. (2014). On Logic Embeddings and Gödel's God. *22nd International Workshop on Algebraic Development Techniques*, (pp. 8-9).

Benzmüller, C., & Woltzenlogel Paleo, B. (2014). Proceedings. In C. Benzmüller, & B. Woltzenlogel Paleo (Ed.), *Eleventh Workshop on User Interfaces for Theorem Provers. 167.* EPTCS.

Benzmüller, C., & Woltzenlogel Paleo, B. (2015). Experiments in Computational Metaphysics. *9th All-India Students' Conference on Science and Spiritual Quest* (pp. 23-40). Kharagpur: Science and Spiritual Quest - Bhaktivedanta Institute.

Benzmüller, C., & Woltzenlogel Paleo, B. (2015). Interacting with Modal Logics in the Coq Proof Assistant. *10th International Computer Science Symposium in Russia. 9139*, pp. 398-411. Springer.

Benzmüller, C., & Woltzenlogel Paleo, B. (2016). The Inconsistency in Gödel's Ontological Argument: A Success Story for AI in Metaphysics. In S. Kambhampati (Ed.), *International Joint Conference on Artificial Intelligence.* New York: AAAI Press.

Benzmüller, C., & Woltzenlogel Paleo, B. (2016). The Modal Collapse as a Collapse of the Modal Square of Opposition. *Studies in Universal Logic*.

Benzmüller, C., Weber, L., & Woltzenlogel Paleo, B. (2016). Computer-Assisted Analysis of the Anderson-Hájek Ontological Controversy. *Logica Universalis, 10*.

Boudou, J., Fellner, A., & Woltzenlogel Paleo, B. (2014). Skeptik: A Proof Compression System. *7th International Joint Conference on Automated Reasoning. LNCS 8562*, pp. 374-380. Springer.

de Spinoza, B. (1677). *Ethica: Ordine Geometrico Demonstrata.*

Dunchev, T., Leitsch, A., Libal, T., Weller, D., & Woltzenlogel Paleo, B. (2010). System Description: The Proof Transformation System CERES. *5th International Joint Conference on Automated Reasoning* (pp. 427-433). Edinburgh: Springer.

Fontaine, P., Merz, S., & Woltzenlogel Paleo, B. (2011). Compression of Propositional Resolution Proofs via Partial Regularization. *23rd International Conference on Automated Deduction* (pp. 237-251). Wroclaw: Springer.

Gödel, K. (1970). 1970 Manuscript of the Ontological Argument. *Collected Works: Unpublished Essays and Letters, 3*. Oxford University Press.

Heule, M. J., & Biere, A. (2015). Proofs for Satisfiability Problems. In W. Paleo, & Bruno (Eds.), *All about Proofs, Proofs for All* (pp. 1-22). London: College Publications.

Lebedeva, E. (2012). Expressing Discourse Dynamics via Continuations. *Ph.D. Thesis*. Nancy, Lorraine, France.

Leibniz, G. W. (1710). *Essais de Théodicée sur la Bonté de Dieu, la Liberté de l'Homme et l'Origine du Mal* (E-Book ed.). (E. M. Huggard, Trans.) Project Gutenberg.

Leibniz, G. W. (1956). *Philosophical Papers and Letters* (Vol. 1 and 2). (L. E. Loemker, Ed., & L. E. Loemker, Trans.) Chicago, Illinois, USA: The University of Chicago Press.

Leibniz, G. W. (1999). *Sämtliche Schriften und Briefe* (Vol. 4 (6th Series)). (H. Schepers, M. Schneider, G. Biller, U. Franke, & H. Kliege-Biller, Eds.) Germany: Akademie Verlag.

Leibniz, G. W. (2006). *Sämtliche Schriften und Briefe* (Vol. 1 (2nd Series)). (M. Schneider, H. Schepers, P. Beeley, G. Biller, K.-B. Herma, & S. Lorenz, Eds.) Germany: Akademie Verlag.

Lenzen, W. (1990). *Das System der Leibniz'schen Logik.* Berlin: de Gruyter.

Lenzen, W. (2016). Leibniz's Ontological Proof of the Existence of God and the Problem of "Impossible Objects". (R. Silvestre, Ed.) *Logica Universalis*(Special Issue on Logic and Religion).

Madarász, J., & Székely, G. (2015). A Completeness Theorem for General Relativity. *2nd International Conference on Logic, Relativity and Beyond.* Budapest.

Miller, D. (2015). Foundational Proof Certificates. In B. Woltzenlogel Paleo (Ed.), *All about Proofs, Proof for All* (pp. 150-163). London: College Publications.

Noble, C. (2010). Leibniz's Comments on Spinoza's Philosophy. *Fourth Annual Conference of the Leibniz Society of North America*.

Oppenheimer, P., & Zalta, E. (2011). A Computationally-Discovered Simplification of the Ontological Argument. *Australasian Journal of Philosophy, 89*(2), 333-349.

Paulin-Mohring, C. (2015). Introduction to the Calculus of Inductive Constructions. In B. Woltzenlogel Paleo (Ed.), *All about Proofs, Proofs for All* (pp. 116-133). London: College Publications.

Rushby, J. (2013). The Ontological Argument in PVS. *CAV Workshop "Fun with Formal Methods".* St. Petersburg.

Russell, B. (1958). *The Philosophy of Leibniz*. London: George Allen and Unwin.

Schulz, S., & Sutcliffe, G. (2015). Proof Generation for Saturating First-Order Theorem Provers. In B. Woltzenlogel Paleo, & D. Delahaye (Eds.), *All about Proofs, Proofs for All* (pp. 45-61). London: College Publications.

Siders, A., & Woltzenlogel Paleo, B. (Forthcoming). Variants of Gödel's Ontological Proof in a Natural Deduction Calculus.

Spinoza. [See: de Spinoza, B.]

Urban, J., Sutcliffe, G., Pudlák, P., & Vyskočil, J. (2008). MaLARea SG1 - Machine Learner for Automated Reasoning with Semantic Guidance. *4th International Joint Conference* (pp. 441-456). Springer Berlin Heidelberg.

Wenzel, M. (2015). Interactive Theorem Proving from the Perspective of Isabelle/Isar. In B. Woltzenlogel Paleo (Ed.), *All about Proofs, Proofs for All* (pp. 91-115). London: College Publications.

Woltzenlogel Paleo, B. (2014, December 14). Reducing Redundancy in Cut-Elimination by Resolution. *Journal of Logic and Computation*.