N.G. de Bruijn (1918–2012) and his Road to Automath, the Earliest Proof Checker

FRANCIEN DECHESNE AND ROB NEDERPELT

The mathematical language Automath was conceived in the second half of the 1960s by N. G. de Bruijn (1918–2012) in Eindhoven, the Netherlands. His aim was to design a fully formal version of the common mathematical manner of expression. This implied, for him, that the language Automath should be relatively close to the way mathematicians usually communicate.

Shortly after the first version of the language came to light, research around Automath became concentrated in the so-called Automath project. A major result of this project was the world's first computer-implemented proof checker. Since the early 1990s, the study and construction of proof checkers, and the related proof assistants, have been growing fields of research – with applications in various areas, such as the development of correct software, the generation of proofs in mathematics, the verification of mathematics and software, and the use of the research for didactic purposes.

Today, it is generally accepted that Automath, the vision behind it, and its potentials, was far ahead of its time. The history and effects of this impressive project have been described several times from the viewpoint of the current interest in tools for proof assistance.

In this article, we aim to give an account of how De Bruijn's mathematical work, and especially his way of doing mathematics, incited him to come up with the revolutionary idea of devising a useful formal language as an aid to the "working mathematician." At that time (1967), formalizing mathematics was out of fashion, because K. Gödel had proven, in his (first) incompleteness theorem of 1931, that no formalization containing elementary arithmetic can be both consistent and complete (in the sense that all true statements can be proven within that formalization). Moreover, applying computers to "real" mathematics was regarded with skepticism by the mathematical community, of which he was a well-respected member.

We think two factors were essential in the origin of de Bruijn's vision. First, he was not actively interested in the foundations or philosophy of mathematics, but he had formed distinctive views on the practice of *doing* mathematics. In Automath, we can recognize his own mathematical

DOI 10.1007/s00283-012-9324-x

Published online: 20 October 2012

experiences. The second factor enabling De Bruijn to pursue his vision was his tendency to follow consistently his own route, often against other people's opinions. This strong inclination was a consequence of the intellectual independence he acquired in his youth, leading him to construct his insights from scratch rather than to build on the work of others.

A Concise Mathematical Biography

Nicolaas Govert ("Dick") de Bruijn was born on July 9, 1918, in the Dutch city of The Hague. From 1930 to 1934 he attended the "Hogere Burgerschool," *HBS*, a form of public secondary education in the Dutch school system that has existed for almost 150 years.

Of this period he recalls that he was not very happy with "memory-subjects," things one had to recall without any clear associations, and that he had bad visual memory; once he had to come back after school to memorize the names of stuffed birds because he failed a test on them.

On the other hand, he proved to be a smart student, completing the 5-year program of the HBS in 4 years.

Unfortunately, he was too young to qualify for a university scholarship directly afterward, nor could he find an appropriate job in the Depression years of the 1930s. To bridge the time, he studied independently and solitarily for teaching certificates (K1-K5) in mathematics. One of the books he had to use was by F. Schuh (1875–1966), professor at the Technische Hogeschool Delft (currently Delft University of Technology) [30]. De Bruijn recalled that he constructed the proofs for himself first, and only afterward looked in the book.

In 1936, at the age of 18, thanks to his K5-exam results, he did obtain a scholarship and was able to enroll in the mathematics program of Leiden University. He was particularly inspired by the young lecturer H. D. Kloosterman, who was responsible for large parts of the teaching to first- and second-year students. De Bruijn especially remembered the *caput* lectures on modern topics (such as Lebesgue integrals, linear operators in Hilbert spaces, group theory, and number theory), which Kloosterman built up from the ground. Starting a subject from scratch – that pleased De Bruijn extremely.

De Bruijn and Kloosterman also shared an appreciation for precise formulation. Kloosterman is reported to have remarked that he had hardly ever met a student who formulated more precisely. And conversely, De Bruijn states in his *In Memoriam* for Kloosterman: "[…] from Kloosterman I inherited his love for precision in presentation, and his love for correct mathematical language. Things need not be vague in order to be interesting. It was his style to be careful, precise, clear, patient, right to the goal, never a superfluous word." [12]

From 1939 to 1944, while continuing his studies toward his university graduation and Ph.D. degree in Leiden, De Bruijn supported himself as an assistant at the Technische Hogeschool in Delft. In 1940, however, classes in Leiden were suspended by the German occupiers because of student protests against the discharge of Jewish professors, and just after De Bruijn's preparatory exam for the doctorate, Leiden University stopped awarding degrees. The official thesis defense [7] took place at the Vrije Universiteit in Amsterdam in March 1943, with J. F. Koksma as official promotor. His research had been supervised by Kloosterman, and the results (on modular forms, a topic in number theory) were produced very quickly; he only really started working on this research in August 1942. Under the circumstances, it is understandable that de Bruijn worked mostly by himself.

After obtaining his Ph.D., de Bruijn took a research job at Philips NatLab research center in Eindhoven (1944–1946) before returning to Delft as professor in 1946. In 1952, he was appointed professor at the University of Amsterdam; E. W. Beth and A. Heyting were his colleagues there. Despite the prestige of the University of Amsterdam, which had been the center of Dutch mathematics for decades, J. J. Seidel (1919–2001) managed to lure De Bruijn to the Technische

AUTHORS

FRANCIEN DECHESNE (born 1971) studied Mathematics and Philosophy at the Radboud University Nijmegen, and received a Ph.D. in Logic (2005) in a project at Tilburg University and Eindhoven University of Technology. Her research interests include intuitionistic mathematics, logic, computer security, and the ethical impact of information technology; the latter is the topic of her research at Delft University of Technology. She likes to fill her nonacademic life with activities involving bicycles, music, and words (preferably the combination thereof) together with her friends.

Delft University of Technology
Delft
The Netherlands
e-mail: f.dechesne@tudelft.nl

ROB NEDERPELT (born 1942) was Associate Professor of Applied Logic at Eindhoven University of Technology (The Netherlands) until his retirement. Around 1970 he participated in N. G. de Bruijn's Automath project. He has written many papers and a number of books, often inspired by those early years. His main research interests are type theory, lambda calculus, and the language of mathematics. He enjoys family life, chamber music, and jazz (John Coltrane!), he writes theatre reviews, and he loves to play tennis.

Eindhoven University of Technology
Eindhoven
The Netherlands
e-mail: r.p.nederpelt@tue.nl

Hogeschool Eindhoven (now Eindhoven University of Technology) in 1960. Seidel, De Bruijn's former fellow student and friend at Leiden University, had organized a new mathematics department there. In this fresh and expanding environment, De Bruijn had great freedom in his research topics. The mathematics department grew in eminence; by 1972, four of the ten KNAW (Royal Netherlands Academy of Arts and Sciences) members in mathematics worked in that department (the other three were C. J. Bouwkamp, E. W. Dijkstra, and J. H. van Lint) [6, 13].

De Bruijn was a respected and prolific mathematician throughout his career, with almost 200 articles in journals and proceedings, and (contributions to) several books. He started as a specialist in analytic number theory, which led to an early correspondence with P. Erdös, and personal contacts from 1948 onward. In those years, he found a paper by G. Pólya in the collection of L. O. Blumenthal in the extensive mathematical library of Delft University, and he began corresponding with Pólya. This led to personal contact, cumulating in the so-called Pólya-de Bruijn enumeration, a famous result in combinatorics.

Gradually de Bruijn's interest grew wider to include combinatorics, asymptotics, function theory, optimal control, Fourier theory, type theory, and quasicrystals. De Bruijn's name has been attached to several mathematical notions, such as de Bruijn cycles and graphs, de Bruijn-Newman constant in Fourier theory, one Erdös-de Bruijn theorem about coloring of graphs and another about finite geometry, de Bruijn indices and de Bruijn notation in typed lambda calculus, and the de Bruijn criterion in type theory. (See the references from [32] onwards for de Bruijn's own assessment of the highlights of his work.)

A Tendency to a Clean Formulation

De Bruijn always focused on *doing* (and teaching) mathematics rather than reflecting on it. In presenting his mathematical results, he became more and more convinced of the crucial importance of a transparent exposition and an accurate formulation – both for the writer and for the reader. So he became interested in the question of what the basic structures of mathematics were.

Foundations had hardly been mentioned in the curriculum in Leiden, but it was a central theme in Amsterdam, especially when De Bruijn worked there. One would expect that having some famous descendants of L. E. J. Brouwer as colleagues, notably the intuitionist A. Heyting and the logician E. W. Beth, De Bruijn's attention – so much focused on proofs, precision, and the "nature" of the mathematical craft – could have been drawn toward logic and the foundations of mathematics. But instead of moving toward meta-mathematics, he produced some of his most influential mathematical works during his Amsterdam period, including his book *Asymptotic Methods in Analysis* [32].

In his work on analysis, de Bruijn tried to avoid the axiom of choice (and in general he avoided set theory, as it did not reflect mathematical practice). Heyting is reported to have found it remarkable that someone could care so much about avoiding the axiom of choice while accepting the law of the excluded middle (both are nonconstructive and rejected in Intuitionism), but it was only after working on Automath that De Bruijn himself understood that there was a parallel between the two [26, A7, p. 205].

It is interesting to note that, on the other hand, he incorporated a short introduction to the language of modern mathematics, including elementary logic, in the first weeks of his Amsterdam course in Linear Algebra, because it would train the students in proper formulation and reasoning. We see this as an example of De Bruijn's general attitude toward logic and foundations of mathematics: he taught the logic as far as – in his opinion – the students needed it for *doing* mathematics. For de Bruijn, logic was not an object of study in its own right, but a means to represent the basics of mathematical reasoning and the standard of precision required for that.

De Bruijn's Early Interest in Computers

De Bruijn was interested in mechanical calculators in his HBS-years, and he remained actively interested in the developments leading to the first computing machines. When these became available, he immediately took the opportunity to use them in support of his mathematical work. In this respect, he differed from many of his fellow mathematicians who did not consider work in this field as relevant for mathematics. Kloosterman, for instance, was very suspicious toward computers and programming.

At Philips NatLab, de Bruijn heard about the electronic ENIAC computer, developed for the U.S. army. It was said to be able to calculate the track of a dropped bomb faster than the bomb could hit the ground. This inspired him to invent a mechanical machine with marbles running through it, aiming to perform the same logical operations as the ENIAC with its vacuum tubes, but just a bit slower.

In the same period, he also designed an electronic binary adding machine that added *directly* (i.e., without taking the consecutive bits one by one). The idea was not followed up; it turned out that a similar technique was already implemented in a machine created by Bell Labs in the United States.

During his professorship in Delft, for the optics professor A. C. S. van Heel, de Bruijn was involved in the development of an automatic calculator for the radius of lenses. This idea was executed in 1950 by W. L. van der Poel.

In the 1950s, A. van Wijngaarden, later one of the central figures in the ALGOL project, presented a talk in Delft on J. von Neumann's idea of using one and the same memory for data and instructions. This idea intrigued De Bruijn and resulted, decades later, in his insight on the human brain: people use their brains for memories as well as for thinking about these memories (see [10]).

In the early 1960s, Eindhoven University of Technology obtained its first computer, an IBM 1620, for which De Bruijn wrote a little program called *Wouwel* (i.e., "Chatter"), which took a text in an arbitrary language as input, and, on the basis of a statistical analysis of letter combinations, generated a new text as output. The program was written purely in machine language.

In March 1963, de Bruijn programmed a production of the 2339 solutions to the combinatorial Pentomino problem for a 6×10 rectangle ([9], see also [23]). To make this work at reasonable speed, he designed a very long program with many repetitions. Then he wrote a second program that produced that long program, all in machine language. De Bruijn continued to program all kinds of combinatorial problems, first in ALGOL 60, later in PASCAL.

Using ALGOL 60 gave him ideas about the structure of mathematical language. Crucial in this respect was a presentation by A. van Wijngaarden in Eindhoven in 1963 about ALGOL, with its block structured programs. De Bruijn noticed the same block structure in mathematical literature: one uses distinctive block openers ("Assume," "Let"). However, the blocks are hardly ever explicitly closed. (De Bruijn remarked that people tend to send marriage announcements, but not divorce announcements. His Ph.D. student, R. M. A. Wieringa, remembered this and sent cards around when he was divorced.)

It is also worth noting that van Wijngaarden, when he first heard about Automath, responded: "Oh, but it's a language!" (not a program). De Bruijn had not looked at it that way, but agreed.

A Pragmatic Attitude Toward Mathematics

One would expect – also with a view to the history of mathematics – that the wish to formalize mathematics would be closely connected to a *formalist* attitude toward the nature of mathematics; that is, the view (advocated prominently by D. Hilbert) that mathematics essentially *is* the manipulation of meaningless symbols, and not *about* some world of abstract objects.

However, De Bruijn's motivation to design a formal language for mathematics did not come from any such strong view of the nature of mathematics itself. In fact, he never really took a position with respect to what he considered the *true* nature of mathematics, because it was not really important for the day-to-day mathematical practice. Instead, by his extensive experience in *doing* mathematics, he developed a view of how *mathematicians work*. This way of working is common for all areas of mathematics and is independent of any foundational viewpoint. In his conception, *language* is the only way to communicate mathematics.

After reading the textbooks of Fred Schuh [30] while studying for the "aktes" (teaching diplomas), De Bruijn entered new areas of mathematics by starting from the central concepts of the field only and attacking the interesting questions from there (rather than first studying the literature to learn about existing results). This made it essential that definitions be stated clearly and precisely.

This does not mean that mathematics was nothing more than language to de Bruijn. He had a well developed mathematical intuition, which he appreciated highly: in his inaugural speech for his professorship in Delft (1946) [8], he mentioned, as the main source of happiness in doing mathematics, the *Eureka* moments – the sudden insights that resolve a long and sometimes desperate quest. But sometimes intuition fails to lead the way, and then a path forward may be offered through the language. De Bruijn remembered noticing the power of language when he learned about large cardinals while reading Kamke's *Theory of Sets* [22] in 1937; he was amazed at how language can talk coherently about things that make no sense (at least not to him). He also accepted the notion that the linguistic structure of the text may even *guide* the mathematician in constructing proofs, as the following experience demonstrates. This experience also served as an important step toward the conception of Automath.

De Bruijn was working in Eindhoven with his Ph.D. student W. van der Meijden on eliminating the axiom of choice from the proof of a theorem on commutative Banach algebras. The central lemma, involving point-free topology, was very difficult to grasp conceptually, even with a mathematical intuition as trained as De Bruijn's. So, he took a piece of paper of about half a square meter in area, wrote the known elements as logical formulas in the left upper corner, the goal in the lower right corner, and tried to close up the gap, working bottom up and top down simultaneously using nested ("Fitch-style") derivation. All steps were driven almost mechanically by the shape of the "available" logical formulas and the shape of the actual goal, and they did not require understanding the concepts involved. And it worked! The gap was closed, the proof just fit on the piece of paper. De Bruijn then realized that it should be possible, and even feasible, to perform such a mechanical proof with computer assistance.

As a result of all this, De Bruijn became more and more convinced that we do mathematics within a structured language and that, hence, the common language of mathematics is central to mathematical practice. This language, as traditionally employed in mathematical books and papers, is an intelligent mixture of (stylized) natural language and mathematical symbols or formulas. He therefore expected that texts written in the common mathematical language can gain precision by *formalization*: translation into a (completely) formal language. De Bruijn envisaged a far-reaching consequence: when employing a sufficient amount of formal rigor, the question of the *mathematical correctness* of a text in the common language can be reduced to the question of *syntactical correctness* of the text in the formal language. More specifically, the correctness of a mathematical proof can be verified by checking syntactical dependencies between expressions in the formal language translation of that proof. This form of checking can be performed "mechanically," that is, without insight or intuition. From his experiences with computers and programming, De Bruijn was convinced that the checking of formalized proofs could be performed by means of a simple computer program.

The reliability of the automatic verification depends then on the correctness of the computer program that performs the syntactic checking. De Bruijn's strong intuition was that the correctness of such a simple program would be fundamentally easier to establish than the correctness of the proofs that it verifies.

The Automath Project

In 1967, De Bruijn had designed a first version of a formal language for mathematics: the *Primitive Automatic Language* (PAL; [26], A7, p. 210). It already contained most of the basic ideas for such a language. A *book* in PAL was formed by *lines* of text organized in nested blocks. Lines could be either axioms, type declarations for variables, or definitions. Surprisingly, with this simple structure one can represent real pieces of mathematics: basically everything that does not exploit the modern notion of *function*. In fact, in retrospect PAL can be seen as the definition system present in the Automath languages.

By extending PAL in 1968 with the λ-notation for functions, which De Bruijn became acquainted with through the analysis courses of his colleague H. Freudenthal of Utrecht University, he incorporated the modern notion of function as well. This resulted in basic Automath (later rebaptized AUT-68).

The research stimulated by the design of the Automath language and some of its variants became bundled in the so-called Automath project, in which a number of researchers joined forces. The Automath project was initiated and was led by De Bruijn in Eindhoven from 1967 until the early 1980s. Readers interested in a more complete historical and scientific account of the project are referred to the collection *Selected Papers on Automath*, [26], and to the Automath Archive [2]. Both include several retrospective articles by De Bruijn and others involved in the project. Some recent workshops containing historical reviews are *Thirty Five Years of Automating Mathematics* [21], and *Mathematics, Logic and Computation* [25].

De Bruijn's central aim with the project was the design of a formal language in which any piece of mathematics can be straightforwardly expressed, with the property that a computer can check the correctness of the text. The use of the system should be close to mathematical practice and should not depend on any foundational paradigm for mathematics. The project aimed beyond showing that such a thing was theoretically possible: De Bruijn was convinced it was actually feasible (even with the computers of the time), and he was anxious to show this. His dream was that within one or two decades, mathematicians would have a proof checker on their desk that they would actually use for verifying and archiving results (e.g., in the reviewing process), and which possibly also would assist the mathematician in proving new results.

These goals were quite different from other projects in the emerging field of reasoning with computers, as De Bruijn himself noticed at the conference on Automatic Deduction in Versailles, France (1968). The vast majority of these projects were more ambitious: automated theorem proving, i.e., automatic *generation* of proofs, rather than the humbler task of *verification* of proofs that already have been delivered somehow. The formal languages in the proof-generating ventures were necessarily of restricted and specialized expressive power so as to keep the system decidable. Therefore they would never be able to capture entire areas of mathematics, as De Bruijn desired.

Essential Features of Automath

The basic idea behind Automath can be summarized as follows: In the process of "doing" mathematics, the mental process and its expression in written form influence each other beneficially. The conveyance of a (mathematical) thought, even an unfinished one, into written language is a way to make the mathematics clearer and more transparent. On the other hand, a written text of mathematical nature, also with gaps in it, helps the mind to grasp the mathematical content and generates ideas about how to proceed. De Bruijn realized that the linguistic structure of the common way of expressing mathematics is tradition-based, and that, at the time, there existed no coherent formal system of expressions and rules underlying the written "mathematical language." In his view, formalization of mathematics starts with the recognition of the "linguistic" aspects that are specific for mathematics, for example: the definition mechanism, substitution, the role of bound variables, parameters, function abstraction. The presence of a coherent formal apparatus to express these matters, a so-called "logical framework," is all-important and sufficient. The choice of how to formalize logic or set theory can be made when *using* the system in a later stage.

Although de Bruijn envisaged many applications of such a formalization of mathematics for the future, initially he concentrated on the *verification* of mathematical theories. He thought this to be a viable and feasible way to test his ideas and to experiment with different versions in a real mathematical setting.

In the process of testing the basic Automath language AUT-68 for the formalization of several fields in mathematics, De Bruijn and his associates developed a family of Automath languages (AUT-QE, AUT-SL, AUT-Π, etc.), subtly differing from each other in expressivity.

In all Automath variants, the extension with λ-calculus-like expressions was incorporated in order to deal with abstract functions. This enabled the "full" expressivity of the Automath languages. De Bruijn conjectured in retrospect that the unease of mathematicians with the λ-notation may have distanced them from Automath – especially because at that time many of them were raised in the Bourbaki style, which leaned heavily on a set-theoretic approach to mathematics, not on the function concept.

A crucial insight of De Bruijn links the formal Automath languages with the mathematical and logical reasoning it is intended to formalize: types in typed λ-calculus can also be considered as *proof classes*, by regarding the elements of a type as *proofs* of a proposition. This makes it possible to do logic in a typed language, even in the simple language PAL. In his own reconstruction, De Bruijn suggested he received the inspiration for this insight from the functional interpretation of the logical implication, which he learned from the intuitionistic logician A. Heyting [26, A.7, p. 211]. This functional interpretation does not treat $A \rightarrow B$ as equivalent to $\neg A \vee B$, but defines an implication $A \rightarrow B$ to be true if there is a *method* that, given any proof of A, provides a proof of B. Such a "method" resembles the notion of "function" in λ-calculus. This is a stronger notion of implication than the usual *material implication*. As it turned out, this insight had been found by H. B. Curry and W. Howard [19]. Nowadays it is known as *Propositions-as-Types*, *Proofs-as-Terms*, or as the *Curry-Howard(-de Bruijn) isomorphism*.

Since in Automath languages, the syntactic correctness of a text entails the correctness of the mathematical theory it expresses, verification of a text boils down to checking the proper use of the rules of the formal language. That it actually *worked* in practice was convincingly demonstrated in October 1975 (Figure 1), by the crowning achievement of the Automath project: the verification of a formalization of the entire *Grundlagen der Analysis*, a standard textbook by

Figure 1. The last theorem of Edmund Landau's book 'Grundlagen der Analysis' is entered. From left to right: Bram Kornaat, Bert van Benthem Jutting, Ids Zandleven, Roel de Vrijer, N.G. de Bruijn. Source: The Automath Archive [2].

E. Landau [24]. The verification by a Burroughs B6700 computer [5] at Eindhoven University of Technology involved a formal text of 13,433 lines.

The program code for the verification is significantly shorter and simpler than the formal translation of the mathematical text: the translation of the *Grundlagen* was about 500 pages of Automath text, whereas the code of the verifying program was only a few pages long. For an example of the Automath texts, we reproduce in Figure 2 a small piece of the formal text, taken from L. S. van Benthem Jutting's thesis [5, p. 95]. It is a translation into AUT-QE of a theorem concerning natural numbers and the theorem's proof. We include an explanatory commentary.

The Legacy of Automath

Besides technical contributions, such as the "De Bruijn indices" (that circumvent the problem of renaming variables; see [11]), the Automath project has made lasting contributions, such as:

- Independence of logic: it suffices (and is also advantageous) to design a logical framework suited to different kinds of logical systems.
- Venturing "beyond Gödel": computers can be useful for mathematics, even if they cannot *solve* all mathematical problems by themselves.
- Enhanced reliability: the feature that all obtained formal expressions can be checked by one fixed and relatively easy algorithm, which can be validated beforehand once and for all (the "De Bruijn criterion").

These ideas influenced other researchers who felt more or less the same incentive to make mathematics more formal. In the decades after 1968, there have been two kinds of followers of de Bruijn's early experiments: logicians, interested in what a proof *is* and how it can be constructed; and computer scientists, who became more and more aware of the fact that *correctness* of software requires some form of verification: do the procedures meet the specification?

In logic, Automath has been a source of inspiration for research in so-called typed lambda-calculi (or Type Theory): see the standardizing survey of H. P. Barendregt [3]. Automath had a direct influence on the Calculus of Constructions (Th. Coquand and G. Huet [16]), which became the basis of a very influential *proof assistant*, called Coq [15]. A proof assistant is more than a proof checker: it is a computer program that helps the user in constructing formalized proofs, by offering so-called "tactics" that take the proof further. All of this in such a manner that the proof composer keeps the overview (and the computer fills in the necessary, but often uninteresting, steps). For more facts about proof assistants, see [4]. With the aid of Coq, an impressive number of mathematical theories have been formalized and verified, and the quantity is still growing. A proof assistant comparable to Coq, and also influenced by Automath, is Agda [1]. Other systems directly influenced by de Bruijn's ideas of a "logical framework" are LF [18] and TWELF [31]. The intuitionistic type theory of P. Martin-Löf [27] is also inspired by Automath.

Comparable systems, also for verification of (claims about) computer programs, have been developed in the last decades. We mention LCF [28], HOL [17], Isabelle [20], NuPRL [14], and the popular PVS [29]. All these systems are, more or less, outgrowths of Automath.

De Bruijn's original vision, that today all mathematicians would happily use computers to structure their mathematical thoughts on the basis of some Automath-like language, has not materialized. On the contrary, the willingness of the average mathematician to accept computer help for his mathematical proofs is still very small. On the other hand, it becomes more and more conceivable that a computerized "library" of ready, formalized, usable mathematical knowledge, will have enough attraction to become a standard tool on the desk. And once that bridge has been crossed, de Bruijn's dream comes into sight.

Conclusion

Just as de Bruijn attacked problems in new areas of mathematics by starting from scratch, he asked and answered the question of whether it would be possible to design a language in which all of mathematics can be expressed and mechanically verified. In this sense, the Automath project was like his other mathematical endeavors, with the difference being that this time he was assisted by a team of people over several years.

The Automath project demonstrated that automatic verification of mathematics was feasible, even with the computer technology of the early 1970s. However, the achievements of the project were not appreciated at the time. Only two decades later, De Bruijn's work was picked up by a new generation of researchers. They recognized the valuable insights underlying the Automath project, and they applied them in their own work. De Bruijn's pioneering ventures with Automath are nowadays widely appreciated, in particular in the community of applied logicians working in type theory and in verification of mathematical theories and computer programs.

It was de Bruijn's independence of mind that enabled him to pursue a project that rowed against two streams: the skepticism of mathematicians toward computers, and the

```
     *  X        :=   - - -                                      ;  NAT
X    *  Y        :=   - - -                                      ;  NAT
Y    *  PROP1    :=  NIS(Y, PL(X, Y))                            ;  PROP
X    *  T1       :=  SYMNOTIS(NAT, ⟨X⟩SUC, 1, ⟨X⟩AX3)            ;  NIS(1, ⟨X⟩SUC)
X    *  T2       :=  TH4(NAT, 1, ⟨X⟩SUC, PL(X, 1), T1, SATZ4A)   ;  PROP1(1)
Y    *  P        :=   - - -                                      ;  PROP1(Y)
P    *  T3       :=  SATZ1(Y, PL(X, Y), P)                       ;  NIS(⟨Y⟩SUC,
                                                                    ⟨PL(X, Y)⟩SUC)
P    *  T4       :=  TH4(NAT, ⟨Y⟩SUC, ⟨PL(X, Y)⟩SUC,
                     PL(X, ⟨Y⟩SUC), T3, SATZ4B)                  ;  PROP1(⟨Y⟩SUC)
Y    *  SATZ7    :=  INDUCTION([Z, NAT]PROP1(Z), T2,
                     [Z, NAT][U, PROP1(Z)]T4(Z, U), Y)           ;  NIS(Y, PL(X, Y))
```

Comment:
– This text fragment consists of 9 lines of the form $A * B := C\,; D$. The intended meaning of such a line is: in the context ending in A, the name B is given to the expression C, having type D.
– If $C \equiv ---$, then the meaning is slightly different: B is introduced as a new *context variable* of type D (in the context ending in A).
– The first entry of any line takes care of the context administration in a cumulative manner: it mentions the final variable of the intended chain of context variables. In the third line, for example, this parameter chain is X, Y, whereas in the fourth line it is only X. In the seventh line it is X, Y, P.
– Some of the types represent sets and propositions. A confusing fact in this example is that the final symbol PROP (line 3) represents the *collection* of all propositions, whereas PROP1, being of *type* PROP, is a mere proposition.
– It is not difficult to guess that the intended meaning of PROP1 is: $\mathrm{Y} \neq \mathrm{X} + \mathrm{Y}$. See line 3. The final line contains a *proof* that this proposition holds for positive natural numbers X and Y.
– The rest of the text fragment serves to build up this proof by the use of earlier results. The proof is by *induction* (cf. line 9) on Y: first PROP1(1) is proved (lines 4 and 5), and then, assuming PROP1(Y) (line 6), a proof of PROP1(⟨Y⟩SUC) is derived (line 8).
– For better understanding, we mention two technical details: (1) Lambda-abstraction is denoted by square brackets. For example, in the final line, the expression [Z : NAT]PROP1(Z) may be read as the *predicate* corresponding to PROP1, when abstracted over its second context variable. (We do not explain in detail why the *second* variable is involved here.) (2) Lambda-application is denoted by obtuse-angular brackets, *in front of* the "function." Example: ⟨X⟩SUC may be read as SUC "applied to" X (meaning: the successor of the positive natural X).
– The rendered text is part of a much bigger one, as described in Jutting's thesis. Hence, one finds constants here that have been introduced before, such as NIS, PL, SYMNOTIS, NAT, SUC, and AX3. For example: AX3 expresses the third Peano axiom for the positive naturals, stating that the successor of a number cannot be equal to 1. The constant SYMNOTIS names a proof that, if $m \neq n$, then also $n \neq m$. We shall not explain the meaning of the other "imported" constants, as we only intend to give an impression.

Figure 2. A piece of Automath text from van Benthem Jutting's thesis.

focus of the automated reasoning community on theorem proving rather than proof verification. He could row against these streams because he was never the type to "go with the flow."

ACKNOWLEDGMENTS

We are grateful to the recently deceased N. G. de Bruijn for his willingness to discuss the contents of this article. We also thank H. Geuvers, M. Senechal, F. W. Steutel, and the anonymous referee for their helpful comments.

REFERENCES

[1] Agda. http://wiki.portal.chalmers.se/agda/.

[2] The Automath Archive. Hosted at Eindhoven University of Technology: http://www.win.tue.nl/automath/.

[3] H. P. Barendregt. Lambda Calculi with Types. Chapter 2 (pp. 117–309) of S. Ambramsky *et al*. (eds.): *Handbook of Logic in Computer Science*, Vol. 2, Oxford University Press, 1992.

[4] H. Barendregt and H. Geuvers. Proof-Assistants using Dependent Type Systems. Chapter 18 (pp. 1149–1238) of J. A. Robinson and A. Voronkov (eds.): *Handbook of Automated Reasoning*, Vol. 2, Elsevier and The MIT Press, 2001.

[5] L. S. van Benthem Jutting. *Checking Landau's "Grundlagen" in the Automath System*. Ph.D. thesis, Technische Hogeschool Eindhoven, 1977.

[6] Aart Blokhuis and Jack van Lint. In memoriam: Johan Jacob Seidel. *Nieuw Archief voor Wiskunde*, 5/2(3):207–209, September 2001.

[7] N. G. de Bruijn. *Over modulaire vormen van meer veranderlijken*. Ph.D. thesis, Vrije Universiteit Amsterdam, 1943.

[8] N. G. de Bruijn. *Eenige beschouwingen over de waarde der wiskunde (''Some observations on the value of mathematics'')*. Inaugural speech as professor of pure and applied mathematics and theoretical mechanics at Delft University of Technology, 1946.

[9] N. G. de Bruijn. Programmeren van de Pentomino Puzzel (''Programming the Pentomino Puzzle''). *Euclides*, 47:90–104, 1971/1972.

[10] N. G. de Bruijn. *Wiskundige modellen voor het levende brein (''Mathematical models for the living brain'')*. Report of the common meeting of the physics section of the KNAW, 83, No. 10, 1974.

[11] N. G. de Bruijn. Lambda calculus with namefree formulas involving symbols that represent reference transforming mappings. *Indagationes Math*. 40:348–356, 1978.

[12] N. G. de Bruijn. Remembering Kloosterman. *Nieuw Archief voor Wiskunde*, 5/1(2):130–134, June 2000.

[13] N. G. de Bruijn. Jaap Seidel, a friend. *Nieuw Archief voor Wiskunde*, 5/2(3):204–206, September 2001.

[14] R. L. Constable et al. *Implementing Mathematics with the Nuprl Development System*, Prentice-Hall, NJ, 1986.

[15] The Coq Development Team. The Coq Proof Assistant, Reference Manual, Version 8.3. http://coq.inria.fr/refman/.

[16] Th. Coquand and G. Huet. The calculus of constructions. *Information and Computation*, 76:95–120, 1988.

[17] M. J. C. Gordon, T. F. Melham. *Introduction to HOL: A theorem proving environment for higher order logic*. Cambridge University Press, 1993.

[18] R. Harper, F. Honsell, and G. Plotkin. A framework for defining logics. In *Proceedings Second Symposium of Logic in Computer Science*, Ithaca, N.Y., pp. 194–204, IEEE, Washington, DC, 1987.

[19] W. A. Howard. The formulas-as-types notion of construction. In: J. P. Seldin, J. R. Hindley (eds.): *To H. B. Curry: Essays on Combinatory logic, Lambda calculus and Formalism*. Academic Press, pp. 479–490, 1980.

[20] Isabelle. http://isabelle.in.tum.de/.

[21] F. Kamareddine. *Thirty-five years of automating mathematics*. Workshop. Kluwer Academic Publishers, Dordrecht, Boston, 2003. ISBN 1402016565.

[22] E. Kamke. *Theory of Sets*. Dover Publications, 1950. Re-edition.

[23] D. Knuth. *Dancing Links*. Paper P159 at http://www-cs-faculty.stanford.edu/~uno/preprints.html.

[24] E. Landau. *Grundlagen der Analysis*. First ed.: Leipzig, 1930. Third ed.: Chelsea Publ. Comp. New York, 1960.

[25] Mathematics, Logic and Computation. A satellite workshop of ICALP03, in honour of N. G. de Bruijn's 85th anniversary. 4–5 July 2003. http://www.cedar-forest.org/forest/events/Bruijn03/.

[26] R. P. Nederpelt, J. H. Geuvers, and R. C. de Vrijer (eds.): *Selected Papers on Automath*, volume 133 of *Studies in Logic and the Foundations of Computer Science*. Elsevier, 1994.

[27] B. Nordström, K. Petersson, and J. M. Smith. *Programming in Martin-Löf's Type Theory*. Oxford University Press, 1990.

[28] G. Plotkin. LCF considered as a programming language. *Theor. Comp. Sci*. 5:223–255, 1977.

[29] PVS. http://pvs-wiki.csl.sri.com/index.php/Main_Page.

[30] F. Schuh. *Leerboek der Elementaire Theoretische Rekenkunde* (''Textbook on Elementary Theoretical Arithmetic''). Noordhoff. Part 1: *De gehele getallen* (''The Integers''), 1919. Part 2: *De meetbare Getallen* (''The Measurable Numbers''), 1921.

[31] The Twelf Project. http://twelf.plparty.org/wiki/Main_Page.

[32] N. G. de Bruijn. *Asymptotic Methods in Analysis*. North Holland Publishing Company and P. Noordhoff, 1958; Dover Publications, Inc., New York, 1981.

[33] N. G. de Bruijn. Ein Satz über schlichte Funktionen (''A theorem on schlicht functions''). *Nederl. Akad. Wetensch. Proceedings*, 44:47–49, 1941. (=*Indagationes Math*., 3:8–10, 1941).

[34] N. G. de Bruijn. On Mahler's partition problem. *Nederl. Akad. Wetensch. Proceedings*, 51: 659–669, 1948. (=*Indagationes Math*., 10:210–220, 1948).

[35] N. G. de Bruijn. The roots of trigonometric integrals. *Duke Math. J*., 17:197–226, 1950.

[36] N. G. de Bruijn. On bases for the set of integers. *Publ. Math. Debrecen*, 1:232–242, 1950.

[37] N. G. de Bruijn. On the number of positive integers $\leq x$ and free of prime factors $> y$. *Nederl. Akad. Wetensch. Proceedings*, 53:813–821, 1950. (=*Indagationes Math*., 12:257–265, 1950).

[38] N. G. de Bruijn. Functions whose differences belong to a given class. *Nieuw Archief Wiskunde* (2) 23:194–218, 1951.

[39] N. G. de Bruijn. Function theory in Banach algebras. *Ann. Acad. Sci. Fennicae Ser. A. I. Math*. 250/5, 1958. 12 pp.

[40] N. G. de Bruijn. Generalization of Pólya's fundamental theorem in enumerative combinatorial analysis. *Nederl. Akad. Wetensch. Proceedings Ser. A* 62: 59–69, 1959. (=*Indagationes Math*. 21).

[41] N. G. de Bruijn. The mathematical language Automath, its usage, and some of its extensions. Symposium on Automatic Demonstration (Versailles, December 1968). *Lecture Notes in Mathematics*, vol. 125. Springer-Verlag, 1970, pp. 29–61. Reprinted in [26], pp. 73–100.

[42] N. G. de Bruijn. A theory of generalized functions, with applications to Wigner distribution and Weyl correspondence. *Nieuw Archief Wiskunde* (3) 21:205–280, 1973.

[43] N. G. de Bruijn. Defining reals without the use of rationals. *Nederl. Akad. Wetensch. Proceedings Ser. A* 79:100–108, 1976. (=*Indagationes Math*. 36).

[44] N. G. de Bruijn. Algebraic theory of Penrose's non-periodic tilings of the plane. *Nederl. Akad. Wetensch. Proceedings Ser. A* 84:38–66, 1981. (=*Indagationes Math*. 43). Reprinted in: *The Physics of Quasicrystals*, P. J. Steinhardt and S. Ostlund (eds.), World Scientific Publ. Comp., Singapore, 1987, pp. 673–700.

[45] N. G. de Bruijn. Can people think? *Journal of Consciousness Studies*, Vol. 3, 1996, pp. 425–447.