THIRTY FIVE YEARS OF AUTOMATING MATHEMATICS: DEDICATED TO 35 YEARS OF DE BRUIJN'S AUTOMATH

N.G. de Bruijn was a well established mathematician before deciding in 1967 at the age of 49 to work on a new direction related to *Automating Mathematics*. By then, his contributions in mathematics were numerous and extremely influential. His book on advanced asymptotic methods, North-Holland 1958, was a classic and was subsequently turned into a book in the well known Dover book series. His work on combinatorics yielded influential notions and theorems of which we mention the de Bruijn-sequences of 1946 and the de Bruijn-Erdös theorem of 1948. De Bruijn's contributions to mathematics also included his work on generalized function theory, analytic number theory, optimal control, quasicrystals, the mathematical analysis of games and much more. In the 1960s de Bruijn became fascinated by the new computer technology and as a result, decided to start the new AUTOMATH project where he could check, with the help of the computer, the correctness of books of mathematics.

In each area that de Bruijn approached, he shed a new light and was known for his originality and for making deep intellectual contributions. And when it came to automating mathematics, he again did it his way and introduced the highly influential AUTOMATH. In the past decade he has also been working on theories of the human brain.

Through his work on AUTOMATH, de Bruijn started a revolution in using the computer for verification, and since his AUTOMATH, we have seen more and more proof-checking and theorem-proving systems. Although now AUTOMATH is mainly of historical interest,[1] its influence remains impressive and its literature [Nederpelt *et al.*, 1994] is indispensable. This is amazing considering that only a handful of people really worked on building AUTOMATH whereas these days tens of people are usually involved in any influential theorem prover or proof checker. Even those who do not do proof checking use many of the notions given to us by de Bruijn during his AUTOMATH project. For example:

- De Bruijn indices [de Bruijn, 1972] still play an important role in the implementation of programming languages and theorem provers.

[1] Freek Wiedijk has resurrected AUTOMATH [Wiedijk, 2002] with a new implementation (called 'aut') of de Bruijn's Zandleven AUTOMATH checker from the seventies. Wiedijk's implementation describes in some detail the features of aut and was written to restore a damaged version of Jutting's translation [van Benthem-Jutting, 1976] of Landau's book [Landau, 1930]. Wiedijk establishes that aut is quite fast, even when compared to current theorem prover systems (aut can check the translation of a full book in under a second).

Fairouz Kamareddine (eds.),
Thirty-Five Years of Automating Mathematics *1–8.*

- De Bruijn's AUTOMATH introduced influential typing notions which inspired new powerful type systems [Coquand and Huet, 1988].

And of course, some of those involved in proof checking or in the foundations of mathematics have put into use many of the lessons learned from de Bruijn's AUTOMATH. As examples, we mention:

- The article of Robert Constable in this volume which illustrates some of the notions that were introduced in de Bruijn's AUTOMATH and which subsequently influenced new theories (e.g., Scott's constructive validity of [Scott, 1970]) and major theorem provers like Nuprl [Constable *et al.*, 1986] and Coq [Dowek *et al.*, 1993].

- De Bruijn's variant of the propositions-as-types (PAT) principle (which arose independently from the work of Curry and Feys and later Howard) and especially the bool-style implementation of this principle, which has been used in the Edinburgh Logical Framework [Harper *et al.*, 1987] and many other systems (cf. the article of Kamareddine, Laan and Nederpelt in this volume).

- De Bruijn's mathematical vernacular [de Bruijn, 1994], his variant notation of the lambda calculus and his system $\Delta\Lambda$ (cf. B.7 of [Nederpelt *et al.*, 1994]) have not yet received the attention they deserve but this is being rectified in [Kamareddine and Nederpelt, 2003; Nederpelt, 2002; Kamareddine and Nederpelt, 1996].

AUTOMATH was written in Algol 60 and implemented on the primitive computers of the sixties. Thirty-five years on, both technology and theory have evolved a lot leading to impressive new directions in using the computer for manipulating and checking mathematics. De Bruijn and AUTOMATH remain a source of inspiration for much research on type theory, logical frameworks, theorem proving and the checking of mathematics.

Both the occasion of thirty-five years since the beginning of AUTOMATH and the eighty-fifth anniversary of de Bruijn are within one year of each other. To celebrate thirty-five years of AUTOMATH and some of the impressive directions in using computers for mathematics, a workshop was held in April 2002 at Heriot-Watt University, Edinburgh, Scotland (see `http://www.macs.hw.ac.uk/~fairouz/automath2002/`). To celebrate the eighty-fifth anniversary of N.G. de Bruijn a workshop will be held in July 2003 at Eindhoven University of Technology where de Bruijn developed his AUTOMATH (see `http://www.macs.hw.ac.uk/~fairouz/Bruijn03/`). Interest in these and other workshops illustrate that the importance of formalizing, mechanizing and automating mathematics is now undisputable and much work is carried out all over the world on this fascinating topic. It has long been an undisputable fact that N.G. de Bruijn is greatly respected and deeply influential and that his ideas will continue to influence many of us

for a very long time. But, it is not only his mind that has influenced us all. He is a challenge to be with, continuously providing puzzles to solve and never allowing the mind to relax even after a very charged day. His strong personality, extreme wit, sharp humour and high energy continue to touch us. Those of us who know him personally, have not only deep respect for him but a lot of love too and feel extremely privileged to have known him and worked with him.

A call for papers related to mechanizing and automating mathematics was sent out after the workshop at Heriot-Watt in April 2002. Accepted papers from that call appear in [Kamareddine, 2002].

A special volume with dedications for de Bruijn on the occasion of his eighty-fifth anniversary is scheduled for 2004.

This present volume is a collection of papers with a personal flavour. The first paper is by de Bruijn himself, the next four papers are by people who were directly influenced by de Bruijn, and the sixth article studies a Hoare logic which conforms to de Bruijn's criteria. The remaining five articles propose interesting variations to or examples of mechanising mathematics and illustrate different developments in the field in the past 35 years. These articles are summarised as follows:

- The article of N.G. de Bruijn is a personal tale of ideas that *try to find principles for the organisation of a big molecular computer like the one we seem to have (or to be) ourselves.* As the title explains, this paper treats mathematical models for biological memory and conciousness and provides ideas to assist in filling the big gap between mind and matter. De Bruijn's work on mind was partly influenced by the discovery of DNA in the fifties and his work on AUTOMATH in the sixties. In his apology at the end of the article he explains how he got to the idea that thinking is jigsaw puzzling and not algorithmic computation. This article illustrates over and over again the originality of de Bruijn and his constantly working mind, always trying to find answers and to solve puzzles.

- The article of Henk Barendregt starts from a plea of de Bruijn to use a mathematical vernacular for formalising proofs. Barendregt proposes a Mathematical Proof Language (MPL) which is between informal mathematics and formalised mathematics with the idea that MPL can be translated automatically into the formalised language of interactive proof assistants. Such a language will help make proof assistants more user friendly and will also open the door to the mathematicians to become themselves users of these proof assistants. These two motivations alone make the development of languages like MPL an urgent task for the community.

- The article of Robert Constable illustrates the influence of two basic ideas from AUTOMATH on important concepts in type theory. In his AUTOMATH, de Bruijn defined *telescopes* (which provide contexts for theorems) and the concepts of *definitional equality* and *book equality* (see the article of Kamareddine, Laan and Nederpelt in this volume). Constable describes how these concepts compare to recent developments in computational type theory made by his Nuprl group. A telescope is internally represented by a dependent record type and is used in Nuprl to define theories. Definitional equality is computational equality whereas book equality is a quotient type. These associations are fascinating and carry de Bruijn's pragmatic insights further in a computational setting.

- The article of Gérard Huet describes two design issues related to fundamental representation structures for symbolic and logic computations. The paper puts together convincingly two techniques which on the surface may seem unrelated: managing contexts (using the so-called zippers) and sharing (using the sharing functor). Zipper structures allow the proper presentation of linear contexts with substructure ordering and have a certain resonance with de Bruijn's proposal for the representation of lambda terms in his $\Delta\Lambda$ (cf. B.7 of [Nederpelt *et al.*, 1994]). Huet establishes that zippers are dual to trees and can also be seen as linear maps over trees. The sharing functor provides a uniform structure with which common expressions can be shared as much as possible. This article is an excellent insight into the design of symbolic computation systems.

- The article by Kamareddine, Laan and Nederpelt, studies the position of the AUTOMATH systems within the framework of Pure Type Systems (PTSs). In [Barendregt, 1992; Geuvers, 1993], a rough relationship has been given between AUTOMATH and PTSs. That relationship ignores three of the most important features of AUTOMATH: *definitions*, *parameters* and Π-reduction, because at the time, formulations of PTSs did not have these features. Since then, PTSs have been extended with these features and, in view of this, this article revisits the correspondence between AUTOMATH and PTSs. This paper gives the most accurate description of AUTOMATH as a PTS so far.

- The article by Michael Franssen from Eindhoven University of Technology presents several ideas to design a Hoare Logic conforming to de Bruijn's criteria (i.e., a logic whose derivations are mechanically checked). Franssen observes that the standard Hoare Logic includes a rule of consequence which requires the proof of theorems. In order to automate this rule, he extends the Hoare Logic with a typed λ-calculus where proofs of the required theorems are encoded by terms

of the calculus. With this approach, derived programs can be checked once they are constructed and the Hoare Logic can be specified as a type system. Checking whether a program meets it specification becomes a process of type checking. Combining a Hoare logic with a typed λ-calculus enables a reliable tool for deriving correct programs.

- The article of Arnon Avron argues that for the purpose of automated reasoning, there is an interesting logic, somewhere strictly between first and second order logic, determined essentially by an analysis of transitive closure, yielding induction. Avron argues for the special role of the transitive closure operation for understanding inductive definitions and inductive reasoning. He proposes to focus on a logic obtained from first order logic by adding an operator for defining the transitive closure of any defined relation. Avron shows that the transitive closure operation together with a pairing function are enough to generate anything that can be generated by the finitary inductive definitions.

- The article of Randall Holmes presents a formal treatment and reports on an initial implementation of the ramified type theory RTT used in Russell and Whitehead's Principia Mathematica [Russell and Whitehead, 1967]. The article builds on an earlier formalization of RTT in [Kamareddine *et al*, 2002]. As Principia Mathematica was not fully formalised and as many of Russell's and Whitehead's intuitions needed to be interpreted in any formalisation of Principia, there were inevitably places where the formalisation might not meet the intuition. Holmes attempts to capture those places accurately. This paper comes with a different formalisation of Principia based on extracting the theory from the implementation. The Ramified type theory of Russell is motivated. Holmes attempts to remain close to Russell's formalisation whereas [Kamareddine *et al*, 2002] follows the more modern style of type theory.

- The paper of Ruiz-Reina, Alonso, Hidalgo and Martín formalises and proves in the ACL2 theorem prover the well known theorem which states that the multiset relation induced by a well founded relation is also well founded. ACL2 has a restrictive, quantifier-free, first order logic. It is not usually used for formalising mathematics, but instead it is used in the verification of software and hardware. This paper shows that also non-trivial mathematical theorems can be formalised and proved in a system like ACL2 which has a restricted logic. The formalisation of multisets and their mathematical properties enables the authors to give three examples of increasing complexity which show how multisets can be used to prove non-trivial termination properties: the correctness of a program transformation technique, the termina-

tion of McCarthy's 91-function, and Newman's Lemma. The use of a more restrictive logic means that the formalisation is more difficult, but it also means that automation in the proof may be increased.

- The article of Gabbay presents a generalisation of Fraenkel-Mostowski (FM) set theory theory within higher-order logic, and applies it to model the syntax and operational semantics of Milner's π-calculus. Fraenkel-Mostowski set theory and the higher-order logic FM-HOL allow a natural style of reasoning about fresh names and provide a logical framework in which structural induction and alpha-equivalence can cleanly coexist. FM is shown to handle well the subtle use of binding in the π-calculus. Side conditions on free and bound names are taken care of by mixing quantifiers in a wise manner.

- The article of Siekmann, Benzmüller, Fielder, Meier, Normann and Pollet presents a good overview and motivation of the Ωmega system and discusses three different styles of proof development in Ωmega using the example of the irrationality of $\sqrt{2}$. The first style follows the traditial approach of using tactics, the second uses interactive proof planning whereas the third is based on fully automated proof planning. The core of Ωmega is the proof plan data structure (PDS) which manages proofs at different levels of abstraction. Ωmega provides access to external systems (computer algebra systems, automated theorem provers, model generators and constraint solvers) which can be charged with subproblems that arise during proof search.

All the articles in this volume have been refereed in the usual way where 34 referees (from the list of 84 below who reviewed a larger number of papers) took care of the 11 articles in this volume. I am very grateful to all of the referees for their highly valuable work. In particular, I am grateful to: Andreas Abel, Mark Aagaard, Peter Aczel, Thorsten Altenkirsch, Andrea Asperti, David Aspinall, Mauricio Ayala-Rincón, Franco Barbanera, Andrej Bauer, Gilles Barthe, Stefano Berardi, Stefan Berghofe, Frédéric Blanqui, Roel Bloo, Connor McBride, Sylvie Boldo, Ched Brown, Martin Bunder, Carsten Butz, Paul Callaghan, Venanzio Capretta, Kaustuv Chaudhuri, Horatiu Cirstea, Catarina Coquand, Thierry Coquand, Judicaël Courant, Pierre Courtieu, Anuj Dawar, Wil Dekker, Louise Dennis, Mike Dewar, Gilles Dowek, Jean-Christophe Filliâtre, Herman Geuvers, Juergen Giesl, Erich Graedel, Ferrucio Guidi, Chris Hankin, Thérèse Hardin, John Harrison, Hugo Herbelin, Daniel Hirschkoff, Martin Hoffmann, Patrick Holt, Doug Howe, Marieke Huisman, Paul Jackson, Tudor Jebelean, Manfred Kerber, Assaf Kfoury, Michael Kohlhase, Michael Koneçny, Jean-Louis Krivine, James McKinna, Frédéric Lang, Marina Lenisa, John Longley, Marino Miculan, Eugenio Moggi, César Munõz, Pavel Naumov, Masaki Nakamura, Catuscia Palamidessi, Larry Paulson, Simon Peyton Jones, John Power,

Christophe Raffalli, Silvio Ranise, Julian Richardson, Giovanni Sambin, Konrad Slind, Jan Smith, Ian Stark, Juergen Stuber, Nick Taylor, Laurent Théry, Simon Thompson, Franklyn Turbak, Andrzej Trybulec, Betti Venneri, Daria Walukiewicz, Burkhart Wolff, Wai Wong, Hongwei Xi.

I acknowledge the financial and academically stimulating support of the Mathematical Knowledge Management network, EC FP5 grant IST-2001-37057.

Last but not least, I am grateful for Jane Spurr who handled this volume and all the queries in her usual manner: efficient, professional and very friendly. It is always a pleasure to work with Jane.

Fairouz Kamareddine

July 2003

BIBLIOGRAPHY

[Barendregt, 1992] H.P. Barendregt. λ-calculi with types. In *Handbook of Logic in Computer Science*, pages 117–309. OUP, 1992.

[van Benthem-Jutting, 1976] L.S. van Benthem Jutting. A Translation of Landau's "Grundlagen" in AUTOMATH. Technical report, Eindhoven University of Technology, 1976.

[van Benthem-Jutting, 1979] L.S. van Benthem Jutting. *Checking Landau's "Grundlagen" in the Automath system.* PhD thesis, Eindhoven University of Technology, 1977. Published as Mathematical Centre Tracts nr. 83 (Amsterdam, Mathematisch Centrum, 1979).

[de Bruijn, 1968] N.G. de Bruijn. AUTOMATH, a language for mathematics. Technical Report 68-WSK-05, T.H.-Reports, Eindhoven University of Technology, 1968.

[de Bruijn, 1972] N. G. de Bruijn. Lambda-Calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser Theorem. *Indag. Mat.*, 34(5):381–392, 1972.

[de Bruijn, 1970] N.G. de Bruijn. The mathematical language AUTOMATH, its usage and some of its extensions. In M. Laudet, D. Lacombe, and M. Schuetzenberger, editors, *Symposium on Automatic Demonstration*, pages 29–61, IRIA, Versailles, 1968. Springer Verlag, Berlin, 1970. Lecture Notes in Mathematics **125**; also in [Nederpelt *et al.*, 1994], pages 73–100.

[de Bruijn, 1990] N.G. de Bruijn. Reflections on Automath. Eindhoven University of Technology, 1990. Also in [Nederpelt *et al.*, 1994], pages 201–228.

[de Bruijn, 1994] N.G. de Bruijn. The mathematical vernacular, a language for mathematics and typed sets, in *Nederpelt et al.* [1994], North-Holland, Elsevier, pp. 865–936. 1994.

[Constable *et al.*, 1986] R. L. Constable, S. F. Allen, H. M. Bromley, W. R. Cleaveland, J. F. Cremer, R. W. Harper, D. J. Howe, T. B. Knoblock, N. P. Mendler, P. Panangaden, J. T. Sasaki, and S. F. Smith. *Implementing Mathematics with the Nuprl Development System.* Prentice-Hall, NJ, 1986.

[Coquand and Huet, 1988] T. Coquand and G. Huet. The calculus of constructions. *Information and computation*, 76(2/3):95–120, February/March 1988.

[Curry and Feys, 1958] H.B. Curry and R. Feys. *Combinatory Logic I.* Studies in Logic and the Foundations of Mathematics. North-Holland, Amsterdam, 1958.

[Dowek *et al.*, 1993] G. Dowek, A. Felty, H. Herbelin, G. Huet, C. Murthy, C. Parent, C. Paulin-Mohring, and B. Werner. *The Coq Proof Assistant User's Guide*. INRIA, Version 5.8, 1993.

[Geuvers, 1993] J.H. Geuvers. *Logics and Type Systems*. PhD thesis, Catholic University of Nijmegen, 1993.

[Harper *et al.*, 1987] R. Harper, F. Honsell, and G. Plotkin. A framework for defining logics. In *Proceedings Second Symposium on Logic in Computer Science*, pages 194–204, Washington D.C., 1987. IEEE.

[Howard, 1980] W.A. Howard. The formulas-as-types notion of construction. In [Seldin and Hindley, 1980], pages 479–490, 1980.

[Kamareddine, 2002] Fairouz Kamaredddine (ed). *Special Issue Mechanizing and Automating Mathematics: In honour of N.G. de Bruijn. Journal of Automated Reasoning*, 29:3-4. 2002.

[Kamareddine and Nederpelt, 1996] Fairouz Kamareddine and Rob Nederpelt. A useful λ-notation. *Theoretical Computer Science*, 155:85–109, 1996.

[Kamareddine *et al*, 2002] Fairouz Kamareddine, Twan Laan and Rob Nederpelt, "Types in mathematics and logic before 1940", *Bulletin of Symbolic Logic*, vol. 8, no. 2, June 2002.

[Kamareddine and Nederpelt, 2003] Fairouz Kamareddine and Rob Nederpelt. A refinement of de Bruijn's formal language of mathematics, *Journal of Logic, Language and Information*, to appear.

[Laan, 1997] Twan Laan. *The Evolution of Type Theory in Logic and Mathematics*. PhD thesis, Eindhoven University of Technology, 1997.

[Landau, 1930] E. Landau. *Grundlagen der Analysis*. , Leipzig, 1930.

[Nederpelt, 2002] Rob Nederpelt. Weak Type Theory: a formal language for mathematics, *Technical Report 02-05*, Dept. of Mathematics and Computer Science, Eindhoven University of Technology, Box 513, 5600 MB Eindhoven, The Netherlands. 2002. URL: `<vubisweb.tue.nl/N/scripts/mgwms32.dll?TS=SA>`.

[Nederpelt *et al.*, 1994] Rob Nederpelt, J. Herman Geuvers, and Roel C. de Vrijer. *Selected Papers on Automath*. North-Holland, Amsterdam, 1994.

[Scott, 1970] D. Scott. Constructive validity. In D. L. M. Laudelt, editor, *Symposium on Automatic Demonstration*, volume 5(3) of *Lecture Notes in Mathematics*, pages 237–275. Springer-Verlag, New York, 1970.

[Seldin and Hindley, 1980] J.P. Seldin and J.R. Hindley, editors. *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*. Academic Press, New York, 1980.

[Russell and Whitehead, 1967] Alfred N. Whitehead and Bertrand Russell. *Principia Mathematica (to *56)*, Cambridge University Press, 1967.

[Wiedijk, 2002] F. Wiedijk. A new implementation of Automath. In *Special Issue Mechanizing and Automating Mathematics: In honour of N.G. de Bruijn. Journal of Automated Reasoning*, 29:3-4, 183-418. 2002.