A Portrait of a Scientist as a Computational Logician

Maurice Bruynooghe[1], Luis Moniz Pereira[2],
Jöerg H. Siekmann[3], and Maarten van Emden[4]

[1] Department of Computer Science,
K.U.Leuven,
Belgium
[2] Departamento de Informatica,
Universidade Nova de Lisboa,
Portugal
[3] Saarland University,
German Research Center for Artificial Intelligence (DFKI),
Saarbruecken,
Germany
[4] Department of Computer Science,
University of Victoria,
Victoria, British Columbia,
Canada

Throughout his prolific scientific career, Robert (Bob) Kowalski was motivated by his desire to reshape logic from an abstract mathematical discipline into a working tool for problem solving. This led him towards a wide exploration of logic in computer science, artificial intelligence, cognitive science, and law.

His scientific achievements in these pursuits have become landmarks. To this we should add the enthusiasm and leadership with which he has enrolled into this venture an entire community extending over two generations of researchers.

Below we detail by topic some of his accomplishments.

1 Automated Theorem Proving

Bob's early work was part of the enormous enthusiasm generated by Robinson's discovery of the resolution principle. Bob started off with important technical contributions, with Hayes on semantic trees and with Kuehner on SL resolution. The pinnacle of this line of research is Bob's Connection Graph proof procedure.

Already before the Connection Graph proof procedure, Bob was concerned with the redundancy of unrestricted resolution. He collaborated with workers in operations research applying search techniques to guide resolution theorem-provers.

2 Logic for Problem Solving

A formative episode in Bob's development was the backlash against resolution theorem-proving. Green had shown how goals of plans could be elegantly formulated in logic and that the plans themselves could be read off from the proofs that showed the goals were achievable. On the one hand there was the completeness of resolution that suggested this might be feasible. On the other hand there was the painful fact that no existing resolution theorem-prover could implement this research program. An implicit revolt was brewing at MIT with, for example, the development of Hewitt's PLANNER.

Resolution theorem-proving was demoted from a hot topic to a relic of the misguided past. Bob doggedly stuck to his faith in the potential of resolution theorem proving. He carefully studied PLANNER. He worked with Colmerauer on the representation of grammars in logic, discovering the importance of Horn clauses. In this way it was discovered how proofs could be parses, vindicating part of Green's grand vision according to which proofs could be executions of plans that achieve goals formulated in logic. Thus Logic for Problem Solving was born.

3 Logic Programming

Logic for problem-solving, specifically how to represent grammars in logic and how to parse by resolution proofs, influenced the conception of Prolog by Colmerauer and Roussel. Conversely, Prolog influenced logic for problem-solving so that it spawned a well-defined subset that we now know as logic programming.

The birth of the logic programming paradigm had a great impact. Its elegance, simplicity and generality offered a new perspective on many areas in computer science and artificial intelligence. It resulted in several novel programming languages, led to the development of deductive databases, was the foundation for the influential constraint logic programming paradigm, inspired much innovating work in natural language processing, had great influence on developments within knowledge representation, and was the basis for inductive logic programming, a recent offspring from machine learning.

Bob's influential dictum "Algorithm = Logic + Control" provided fundamental direction for increasing clarity and scope in the description of algorithms and design of new control mechanisms for logic programming languages, namely through meta-programming. His subsequent research revealed the potential of the logic programming paradigm in many areas.

4 Logic across the Children's Curriculum

Bob's research program, born in the dark days around 1971, was vindicated in the programming language area when a prominent member of the MIT AI group said, much later, "Prolog is PLANNER done right". But the research program is more radical: logic is not just a good model for programming languages, but also for the way humans think by nature. To test this wider concept, a project was started at a school in London for a class of children who were about 13 years old. A key ingredient was Micro-Prolog, a version of Prolog that ran on micro-computers (as PCs were then called). This system, at the time a revelation, was developed in Bob's group by McCabe and Clark. Another key ingredient was

Ennals, a school teacher, who was trained by Bob in logic programming. Together they developed a curriculum, which was taught on a regular basis for a year by Ennals, with the children writing and running Prolog programs on computers at the school. It showed that with English-like syntax, Horn clauses can be used by children to support their curriculum material in English, mathematics, geography, and history.

5 Logic and Data Bases

Influenced by the pioneering work of Minker, Gallaire, Nicolas and others on the logical analysis and inference techniques for data bases, Bob provided central insight, as well as numerous technical contributions, for this emerging field, that eventually led to the amalgamation of classical data base theory with knowledge representation formalisms in artificial intelligence, logic, and semantic networks. Together with colleagues, Sadri, Sripada and others, he has established significant landmark contributions in various problems such as the frame problem in logic data bases, data base integrity and temporal databases.

6 Logic Programming and the Law

Is mathematical reasoning just typical for proofs of mathematical theorems or can the inspiring vision of Leibniz, that two philosophers in dispute may settle their differences by coding their arguments into an appropriate calculus and then calculate the truth: "CALCULEMUS" be turned into reality?

Bob, in a team effort with Sadri, Sergot and others, showed that the British Nationality Act as well as other highly formalized legislation can be coded into an enchanced logic programming language — and then computed! This insight spawned an interdisciplinary field, logic and law.

7 The Event Calculus

In 1986, at a time when the program of implementing temporal reasoning using Situation Calculus in classical and nonmonotonic logics continued to struggle with conceptual and computational problems, Bob delivered a seminal contribution to the use of logic-based temporal reasoning. In an attempt to overcome the shortcomings of situation calculus, he and Marek Sergot introduced a new ontological concept, the *event* which is an occurrence of an action bound at a specific time point and location. They developed a theory based on this concept, called *Event Calculus* and implemented it in logic programming. This work was very influential and created quite a debate between supporters of the two approaches. Ironically, about ten years later, different researchers including Bob himself showed a close relationship between the event and situation calculi. The work on event calculus is still influential and is applied in the context of AI-applications such as robot control.

8 Common-sense Reasoning

The naive use of negation in PLANNER and early logic programming was soon replaced by the much deeper insight into the distinction between classical negation and what became known as "negation as failure".

Similarly, the early confusion in expert systems between deduction and abduction led to a more thorough investigation and Bob's collaboration with Eshghi, Kakas, Toni and Fung spawned several papers on this issue. Amongst other things these papers compare abduction with negation as failure and have opened the new area of Abductive Logic Programming. Related to this is also Bob's work, with Dung, Toni and others, on argumentation for formalising non-monotonic reasoning.

9 Logic Modeling of Agents

The recent world-wide interest in agents and their applications was met by Bob with a challenge to the Logic Programming community to hone their tools to the issues raised. He led the way himself, publishing with Sadri, on the balanced combination of deliberation and reaction, integrated into an original IFF agent cycle framework, in which the agent at turns reacts and reasons with limited resources. His work paved the road for the involvement of the logic programming community in the flurry of activity we have today concerning computational logic agents and societies of agents.

10 Conclusion

Bob's inspiring leadership and expertise was widely appreciated and sought after the whole world over. His bold initiative to organise a first Logic Programming workshop in May 1976 laid the foundation for an enthusiastic community of logic programmers. His advisory role in projects such as the Japanese Fifth Generation Computing Systems and in organisations such as DFKI, the German National Research Center for A.I. was deep and very influential. As coordinator of the ESPRIT Basic Research Action in Computational Logic, as participant to its successor, Compulog2, and as founding chairman of the ESPRIT network of Excellence in Computational Logic (CompulogNet), he had an enormous impact on the European logic programming research community. His leadership and drive for quality was an example for many young researchers. Distinctions and prizes from many countries pay tribute to his role: MIT Distinguished Lecture, Honorary Distinguished Alumnus of Phi Kappa Phi at the University of Bridgeport, the "Docente a titulo individuale" from Bologna, the fellowships of AAAI, City and Guilds of London Institute, DFKI, ECCAI, and ACM.

As this volume illustrates, Bob's work has established logic as a tool for problem solving and has a lasting influence in many areas of computer science.