Robert Kowalski: A Short Story of My Life and Work
April 2002 - revised June 2015

Schooldays

I was born in Bridgeport, Connecticut, USA, on 15 May 1941. I went to Saint Michael's, a Catholic primary school in Bridgeport, attached to a Polish parish, but I didn't learn much Polish. My parents would speak Polish when they didn't want us children to understand. I was a good student, but not outstanding. There were 60 children in my class, 17 girls and 43 boys.

I went to Fairfield Prep, a boys-only Jesuit High School. It took a long time to get to school, involving half an hour's walk and two buses, each way. In my second year, my Latin teacher, Father Walsh, trained four of us to compete in Latin sight-translation contests. The task was to translate a previously unseen Latin text into English without a dictionary.

The skill needed for the task was the ability to guess the most coherent English translation of the Latin text, constrained by our limited knowledge of Latin and of the subject matter of the text. Many years later, I learned that the required technique - of generating assumptions to solve problems subject to constraints - is called "abduction". Our team took first prize in New England.

I also started to have an intellectual life outside of school. I started reading Freud, Ruth Benedict and Joad's "Guide to Philosophy". I found these books very exciting, but they undermined my Catholic upbringing. I still believed there had to be a single truth, and I wanted to find out what it was. I also wanted to get away from home and to be free to come and go as I pleased.

University of Chicago and University of Bridgeport

For these reasons, I was attracted to the University of Chicago, and intellectually I was not disappointed. Among the other great ideas to which I was exposed in my first year, I was introduced to mathematical logic; and it seemed to me that it might lead the way to the truth.

I got "A"s in all my subjects, except English writing skills, in which I got a "D". I couldn't understand what was wrong with my writing, but I was determined to improve it. I reasoned that if I could understand and solve the problems with my writing, then I would do even better in my other subjects.

At the beginning of my second year, I began to find the social life at the University of Chicago very difficult. To make matters worse, I was overwhelmed by the assigned reading of Gibbon's "Decline and Fall of the Roman Empire". Reading about the seemingly endless *impedimenta* that the Roman troops had to carry to their battles was the last straw. I left Chicago in November of my second year.

I spent the rest of that academic year trying to find myself. I signed up for an expedition to find gold in Honduras, only to abandon the journey somewhere in Ohio. I worked for half a year in a chemical factory as a quality control inspector.

The following academic year, I enrolled at the University of Bridgeport and commuted, with my brothers, Bill and Dan, from my parents' home in Milford. The University of Bridgeport was easy after Chicago. I decided to major in Mathematics.

I couldn't get a scholarship at first. So I supported myself by working in Peoples Savings bank in the evening, processing paper tapes of the day's banking transactions. I discovered how to cut the time of the work in half, mostly by performing multiple tasks in parallel. But, because I was paid by the hour, I then had the more difficult problem of preventing my pay from also being cut in half.

When I went to the scholarship office to argue my case for getting a scholarship, I was turned down because I didn't participate in the extracurricular life of the University. The fact that I was busy working to support my studies was not deemed to be relevant. I was told that the only solution was to join a student club. But, because I had neither the interest nor the time to join any of the existing clubs, I advertised in the student newspaper to announce the formation of a new club for people who didn't want to belong to any clubs. Soon afterwards, I was awarded a scholarship, which allowed me to quit my job at Peoples Savings and to work full time as a student.

Academically, after getting my scholarship, the best thing about the University of Bridgeport was that it left me plenty of time for independent study. Mostly I studied Logic. My favourite title was "The Meaning of Meaning" by C. K. Ogden and I. A. Richards. I also worked on the problems with my English and started to make big improvements in my writing style.

I took the Graduate Record Examination in Mathematics and scored higher than any previous student at the University of Bridgeport. The comparison isn't completely fair, because I had taken the exam a year earlier, and some of the questions were virtually the same. But it had the desired effect. I won Woodrow Wilson and National Science Foundation Fellowships for graduate study. They published my photograph in the Bridgeport Post.

Stanford and University of Warsaw

I went to Stanford to study for a PhD in Mathematics, but my real interest was Logic. I was still looking for the truth, and I was sure that Logic would be the key to finding it. My best course was axiomatic set theory with Dana Scott. He gave us lots of theorems to prove as homework. At first my marks were not very impressive. But Dana Scott marked the coursework himself, and wrote lots of comments. My marks improved significantly as a result.

Jon Barwise was among the other students entering Stanford as a PhD student that year, in 1963. We were friends, but also competitors. He discovered that Stanford had an exchange program with the University of Warsaw, noted for its work in mathematical logic. We both applied for the program. I got in, but he didn't, because he was judged to be too young.

The exchange program started with an intensive Polish course at the end of the summer. I didn't receive any formal credit for the courses I attended at the University of Warsaw, but I didn't have to take any exams, and I could focus exclusively on the logic courses. I took courses with Helena Rasiowa, Andrzej Grzegorczyk and Andrzej Mostowski.

I spent much of my time on extracurricular activities. I met and visited my Polish relatives, including my grandparents, who lived near the Soviet border. I also met my future wife, Danusia, a student in the Mathematics Department at the University. After only a few months, we got married, in February 1965.

Before going to Poland I had no interest in politics or current affairs. But I had been brought up during the Cold War, and the Jesuits were rabidly anti-communist. I expected Poland to be totally devoid of freedom, and I was surprised that it wasn't nearly so bad. However, I didn't fully appreciate how much worse it had been soon after the war, and how much worse it was in many other countries in the Soviet block. I became much more interested and more educated about such matters after I retired.

When I returned to Stanford at the beginning of the next academic year, I found it hard to convince myself that studying complex variables and recursion theory would lead to the truth, and I was upset by the war that was developing in Vietnam. I became one of the organizers of the protest movement and found my niche dreaming up ideas and convincing other people to put them into action. I had the idea of dropping anti-war leaflets from airplanes. My flatmate, Ray Tiernan, a childhood friend from Bridgeport, organized the bombing campaign.

Ray and I both went up in the first few bombing missions. We practiced over Stanford and other places in the San Francisco area. Our first attempt nearly ended in disaster, when the leaflets got caught in the tail of the plane.

Our goal was to bomb the Rose Bowl football game in Los Angeles. Ray and I worked out an elaborate plan to conceal the registration number on the side of the plane, covering it with a false number, which we would rip off during our getaway in mid-fight. Unfortunately, when we landed to cover the number in the Mohave Dessert, before the game, the plane burst a tire, and we were too late to get to the Rose Bowl in time. We bombed Disney Land instead.

Eventually, Ray was arrested on our last mission, when he went up without me.

Puerto Rico and Edinburgh

I left Stanford in the middle of the academic year. Fortunately, I had taken enough courses to leave with a Master's degree. I applied to teach Mathematics at various universities, mostly outside the United States. I eventually accepted a job as Assistant Professor and Acting Chairman of the Mathematics and Physics Department at the Inter-American University in San Juan, Puerto Rico.

I was excited by the prospect of living and working in Puerto Rico, and I studied as much Spanish as I could before leaving. I don't have very clear memories of the one year I worked in Puerto Rico, but it convinced me that I had to start again and finish a PhD if I wanted my colleagues to take me as seriously as I desired. I applied to several universities in Great Britain. Eventually I accepted the offer of a Fellowship from Bernard Meltzer, Head of the Meta-mathematics Unit at the University of Edinburgh.

In the meanwhile my first daughter, Dania, was born. I left Puerto Rico, knowing less Spanish than when I started, because everyone wanted to practice their English.

We used our savings from Puerto Rico to buy a car when we got to England, and we drove it to Edinburgh, after a detour to Poland and Italy, arriving in October 1967. I remember arriving at the doorway of the Meta-mathematics Unit, and seeing the sign: “Department of Computer Science”. My heart sank. I hated computers, but I decided I would stick it out, get my PhD as quickly as possible, and resume my search for the truth.

Bernard Meltzer was working on the automation of mathematical proofs. Although I wasn’t convinced about the value of the research topic, I was determined not to drop out of another PhD. I was lucky. Alan Robinson, the inventor of resolution, was in Edinburgh spending a year’s sabbatical. He had just finished a paper on semantic trees applied to theorem proving with equality. Pat Hayes, another fresh PhD student, and I studied Alan’s paper in minute detail. A few months later, we both wrote our first research paper, on semantic trees.

I finished my PhD in just over two years; and, with a second daughter, Tania, born in Edinburgh, I was free to start a new life. I decided to look for an academic job in the UK. But it wasn’t as easy as I had hoped. I was eventually interviewed for two jobs – one a Fellowship at Pembroke College at Oxford University, the other a Lectureship in the Mathematics Department at the University of Essex.

I knew I wasn’t going to be offered the Fellowship at Pembroke College when the Master of the College introduced me to one of the Fellows as “Mr. Kowalski from the University of Bridgeport”. I didn’t get the other job either. I had to settle instead for a postdoctoral fellowship in the Meta-mathematics Unit in Edinburgh. My third daughter, Janina, was born that same year.

The best thing about the fellowship was that I had plenty of time to explore my real interests. In those days they were mainly in philosophy of science and epistemology. I remember reading and being influenced by Lakatos’ “Proofs and Refutations”, Nelson Goodman’s “Fact, Fiction and Forecast”, and Quine’s “Two Dogmas of Empiricism”. I didn’t fully appreciate it at the time, but in retrospect I was lucky that Bernard encouraged me to explore these broader interests.

Lakatos documented how the history of Euler’s theorem could be viewed as a repeated cycle of conjectured theorems, attempted proofs, counterexamples, and revised conjectures. My reading of Lakatos reinforced my own reflections about research on automated theorem-proving. It encouraged me in the view that it is both harder and more important to identify whether a theorem is worth proving than it is to prove a theorem, whether or not it is worth it. The downside is that it is easy to make a claim about a supposed theorem that cannot later be justified.

Other developments were attracting attention in the world of Logic and Artificial Intelligence. Attacks against Logic were being launched from MIT, with declarative representations declared as bad, and procedural representations as good. In the face of these attacks, many of the researchers working on logic for theorem proving moved into other areas. But I couldn’t accept the view that Logic was dead.

I had been working on a form of resolution, called SL-resolution, with Donald Kuehner, who had been one of my mathematics teachers at the University of Bridgeport. (I visited Donald on one of my visits back home and convinced him to come to Edinburgh to do his PhD. Like me, he thrived in the British PhD environment.)

SL-resolution uses logic in a goal-oriented way. We pointed this out at the end of our paper, and I set out to convince my colleagues that the goal-oriented approach of SL-resolution reconciles logic with the procedural approach advocated at MIT.

In the summer of 1971, I received an invitation from Alain Colmerauer to visit him in Marseille. He was working on natural language understanding, using logic to represent the meaning of sentences and using resolution to derive answers to questions. He was interested in my work on theorem proving and on SL-resolution in particular. My family and I stayed with him and his family for several days in their small flat. We worked late into the night, discovering how to use logic to represent grammars and how to use theorem provers for parsing. We saw that some theorem provers, like hyper-resolution, behaved as bottom-up parsers and others, like SL-resolution, behaved as top-down parsers.

Alain invited me to visit him again for a longer period of two months the following summer of 1972. It was during that second visit that logic programming, as it is commonly understood, was born.

I have tried to document as best I could our various contributions to that idea in an article published in CACM, 1988, and more recently in a History of Logic Programming, published in 2014. In summary, however, it is probably fair to say that my own contributions were mainly philosophical and Alain's were more practical. In particular, Alain's work led in the same summer of 1972 to the design and implementation of the logic programming language Prolog.

Those were heady days. It was obvious to us that we were onto a good idea. Back in Edinburgh, I was lucky in recruiting a number of converts to the cause, David Warren and Maarten van Emden being the most prominent of the earliest recruits. Initially, Bob Boyer and J Moore were also attracted to the idea, and it led to their subsequent work on proving properties of programs written in Lisp.

Edinburgh at that time was a world-renowned centre of research in Artificial Intelligence, and I benefited from the opportunity to discuss ideas with other researchers, including Alan Bundy, Rod Burstall, Michael Gordon, Donald Michie , Robin Milner and Gordon Plotkin, who were working in Edinburgh at that time, and with visitors such as Aaron Sloman and Danny Bobrow.

We also had visitors who were especially attracted to the logic programming idea. They included Luis Pereira from Lisbon, Sten Ake Tarnlund from Stockholm, Peter Szeredi from Budapest and Maurice Bruynooghe from Leuven. I travelled extensively in Europe, giving talks about the new cause.

Before leaving Edinburgh and before finishing my work on automated theorem proving, I developed the connection graph proof procedure. Ironically, the proof procedure was so efficient that even until now there has been no proof of its completeness and only counterexamples to certain limiting cases. (My experience has been that, if a proof procedure is inefficient, then a theorem will have many redundant proofs. Therefore, the fewer the redundancies, the harder it may be to prove that there exists any proof at all. In other words, the more efficient the proof procedure, the harder it may be to prove its completeness.)

The history of attempts to prove the completeness of the connection graph proof procedure reinforced my conviction that identifying theorems is more important than proving them.

Imperial College

Sometime in 1973 or 1974, I was invited to apply for a Readership in the Department of Computing and Control at Imperial College in London. A British Readership is the approximate equivalent of a tenured Associate Professorship in an American university, with the additional feature of being primarily for research. It was an almost unique opportunity to advance my career, and it had the added attraction of being in London, one of the most cosmopolitan and desirable places to live in the world. I jumped at the chance.

It took about a year to confirm my appointment, partly because there was another strong candidate, and partly because doubts were raised about my suitability for the post. I started in January 1975, and was assigned to teach a course on formal languages and automata theory immediately upon my arrival. I knew next to nothing about the subject, and I had little interest in it. Fortunately, Keith Clark, then working as a lecturer at Queen Mary College in London, was a keen convert to logic programming, and he provided me with guidance for the course. I muddled through, but it was an unhappy introduction to the Department.

However, it wasn't long before I was able to redirect my teaching to the areas of logic, logic programming, and artificial intelligence, which were central to my interests. I had to cheat a little in the beginning, for example by setting the students the problem of writing a Prolog interpreter in Cobol, as a programming exercise in the comparative programming languages course.

My first few years at Imperial College were focused on learning enough of the basics of Computing to do my teaching, writing my book "Logic for Problem Solving" and promoting the cause of logic programming in general. In this latter pursuit, I was especially fortunate in recruiting Chris Hogger and in helping to bring Keith Clark into the Department. I also organised the first Logic Programming Workshop, at Imperial College in 1976.

The book was very hard work, and it seemed to take forever. To make matters worse, in those days I didn't type, and I had to rely entirely on others to do all the typing. The final draft was a camera-ready copy produced on a line printer, using ancient word-processing technology. When I finished I knew it would be a long time before I wrote another book.

I visited Syracuse University for an academic term and collaborated with Ken Bowen on amalgamating object level and meta-level logic programming. Our goal was to combine the two levels for practical applications without introducing inconsistencies. Unfortunately, the results were inconclusive. I continued to work on the application of the amalgamated logic to knowledge representation until the early 1990s.

In 1978 I started a course of logic lessons for 12 year old children at my daughters' middle school. We solved logic problems in Prolog on the Departmental computer, using a pay phone connection. The connection would be lost whenever our coins ran out.

Once we demonstrated the feasibility of teaching logic to children, I succeeded in getting support from the Science Research Council to develop microProlog, a microprocessor implementation of Prolog, for use in schools. The project employed Frank McCabe to do the implementation and Richard Ennals to develop and test the teaching materials.

Perhaps the worst thing about my work in those days was the fact that the MSc. course lasted throughout the summer and deprived me of the opportunity to get away from my normal commitments. Earlier, both when I was a student and when I was a postdoctoral researcher in Edinburgh, I relied upon such opportunities to clear my mind of details and to explore broader intellectual horizons.

The Alvey Programme

Then everything changed. In 1981, MITI in Japan announced the Fifth Generation Project, whose stated goal was to leapfrog IBM in ten years time. The governments of Britain, France and Germany were invited to participate, and logic programming was to play a dominant role. At the time, our group at Imperial College was the internationally leading centre for logic programming, and it was the obvious choice for a British centre to collaborate or compete with Japan.

The British government responded by forming a committee chaired by John Alvey, the Director of Research at British Telecom. The academic community, led by the Science Research Council, formed its own committees to advise the Alvey Committee. I was enlisted along with many others to help draft recommendations for the British response. Although I was not yet a full Professor, I was the most senior academic in Britain arguing the logic programming case.

It was chaos. Academics argued with fellow academics, industrialists argued both with academics and other industrialists - all presided over by the British civil service. We all wanted a slice of the action. Some of us went further, arguing that we should follow the lead of the Fifth Generation Project and focus on logic programming to the detriment of other areas. That was a big mistake.

My position in the Department deteriorated, as I came into conflict with my academic colleagues, who wanted the government to focus on mainstream software engineering and formal methods. It wasn't much better on the national level, where logic programming was seen as a newcomer (and some would say an intruder) on the Computing scene. In the end, by the time the Alvey Committee produced its recommendations, virtually every area of Computing and related Electronics was singled out for special attention, with the exception of logic programming, which received hardly a mention.

The British government decided to decline the Japanese invitation and to go it alone. The "Alvey Programme" was established, and eventually, after much further debate, logic programming was identified, along with all the other areas, as worthy of special promotion. By around 1985, as a result of the Alvey Programme and with a lot of help from Keith Clark, the logic programming group at

Imperial College expanded to approximately 50 people, including PhD students, research assistants, academics and support staff. These were supported by thirteen separate, three-year research grants. The administrative and managerial burden was enormous. For my reward - or consolation - I was promoted to a Professorship in 1982.

My position in the Department and that of the logic programming group were strained. We wanted to establish ourselves as a separate entity, and the Department wanted to keep us in our place. In the autumn of 1987, I took a six-month leave of absence, to get away from it all.

Research

From 1981 to 1987, my professional life was dominated by academic politics. It was not an area of activity to which I was naturally drawn, but an area into which I was pushed by events around me. Inevitably the politics interfered with my research.

Fortunately, I was able to continue to make contributions to research by working with PhD students. I worked with Marek Sergot on the application of logic programming to legal reasoning, and along with several other members of the group, including a new PhD student Fariba Sadri, we investigated the formalisation of the British Nationality Act as a logic program. In the atmosphere of the Alvey era, even this caused controversy: Some of our critics accused us of racism, because it was supposed that the work must have been supported by the British government to further its racist policies. I ended up writing to the Guardian, a national newspaper, to try to clear our names.

Marek and I also worked on the representation of temporal reasoning, developing a calculus of events, in the spirit of McCarthy and Hayes' situation calculus, but focusing on the way in which events initiate and terminate local states of affairs. This work became a major thread of a European Community research project, which explored, among other applications, an application to air traffic flow management. Murray Shanahan further developed the event calculus and featured it in his book about the frame problem.

Fariba and I worked on integrity checking for deductive databases. We developed a proof procedure that uses forward reasoning, triggered by an update, to check that a database that satisfied integrity constraints before the update continues to satisfy the integrity constraints after the update. We also investigated the relationship between integrity constraints in databases and rules and exceptions in default reasoning.

I had hoped, during my six-month leave of absence to work on a second book, which I tentatively titled "Logic for Knowledge Representation". Instead, I worked mainly with another PhD student, Kave Eshghi, on abductive logic programming. We adapted the proof procedure for integrity checking to the problem of checking that abductive hypotheses, generated by logic programs, satisfy integrity constraints. We applied the proof procedure to give an abductive interpretation of negation as failure, as a form of default reasoning, in logic programming.

Just as my leave of absence was ending, I received an invitation from Brussels to discuss the possibility of helping to organise a project involving the main academic groups in the European Community working on logic programming. The resulting project, Compulog, employed Fariba as an academic replacement to do my College work, so that during the period 1989-91 I could work full time as a researcher and the project's coordinator. I continued the research that I started earlier, but with greater focus than before.

Soon after the start of the Compulog project, Fujitsu Research Laboratories, which was one of the main partners in the Japanese Fifth Generation Project, approached the College with a proposal to support research on logic programming in our group. As a result of the subsequent discussions, Fujitsu supported a five-year project, focused on abductive logic programming, during the period 1990-95.

In the beginning, the Fujitsu project supported Francesca Toni, as a PhD student. But, when the first three-year grant for the Compulog project ended, I transferred to the Fujitsu project and extended the leave of absence from my College work.

During the first part of this period, I worked mainly on abductive logic programming with Francesca Toni and Tony Kakas. While trying to give an intuitive explanation of Dung's admissibility semantics for logic programming, we formulated an interpretation of the semantics in terms of arguments defending themselves against attack from other arguments.

Dung generalised and abstracted this argumentation interpretation of the admissibility semantics and applied it to other logics for default reasoning. Francesca and I collaborated with Dung during his several visits to Imperial College, supported first by the Fujitsu project and later by a European Community "Keep in Touch" research project.

Towards the end of the Fujitsu project, Fujitsu encouraged me to investigate the application of logic programming to multi-agent systems. This became the most important turning point in my research since my work on logic programming in 1972.

The biggest surprise, which came out of this work, was the realisation that, as a model of computation and reasoning, logic programming is much more restricted than I had previously realised. Fortunately, our earlier work on integrity checking in deductive databases and on abductive logic programming provided much of what was missing: Integrity constraints and integrity checking provided not only the missing functionality of production rules, but also the additional functionality of commitment rules, prohibitions and obligations. Fariba Sadri joined me in this work.

Back in the Department

When the Fujitsu contract ended, I became slowly reintegrated into the life of the Department. Logic programming was beginning to go out of fashion, and the logic programming group was no longer seen as a threat. Indeed, my own rehabilitation was so complete that, during the period 1994-97, I became a member of a four person Departmental Executive Committee, and was even given the title of "Senior Deputy Head of Department".

I'm not really sure what motivated me to get so involved in the running of the Department. Perhaps I wanted to show that I could rise above the parochial interests of the logic programming group and could help to look after the interests of the Department as a whole.

The Department had both external and internal problems. Externally, we suffered the same fate as many other Computing Departments elsewhere. We were the poor relation of the more established departments, and we were inadequately resourced in comparison. When the College decided it should do more to promote Information Technology, it looked primarily to the Electronics and Electrical Engineering (EEE) Department for its lead.

To some extent, our low standing in the College was partially our own doing, the result of a long history of internal conflicts between competing groups. Perhaps it was because I had once been in conflict with the rest of the Department myself and because I had now made my peace that I was so welcome on the Department's Executive Committee.

I began to find my teaching increasingly tedious. The biggest problem was preparing the examinations. The British examination system is very rigorous, and examination papers have to conform to strict constraints. In particular, they need to be approved by an external examiner, and have to be prepared early enough to leave time for any changes required by the external examiner. As a consequence, questions often need to be set and submitted for approval before a course is halfway through. I found that these constraints increasingly inhibited the spontaneity and enthusiasm I could generate for my teaching.

Head of Department

In November 1996, the then Head of Department was so unhappy with the state of the Department and with our relations with the College that he resigned from his post.   He agreed to stay on as Head until the Rector found a replacement. By the beginning of March 1997, there was still no news from the Rector, and the rumour went around that the Department would be broken up and distributed between the Mathematics and the EEE Departments. In desperation, as Senior Deputy Head of Department, I went to talk to the Rector myself.

My real goal was to return to full time research, to work on my book and to be my own boss. Instead, the Rector invited me to become Head of Department, and I accepted. One reason that I agreed to become Head was that I thought that it would give me the opportunity to apply Logic to the practical problems of the Department.

I planned to try to develop general rules to solve problems that would otherwise involve individual, ad hoc negotiations – such problems as deciding academic workloads, the amount of overheads that should be charged on research grants, and the distribution of overheads between the Department and grant holders. I thought that establishing a clear set of rules that applied to everyone alike, without favour or malice, would take the politics out of decision making.

At first, I looked to the College for examples of best practice. I found a number of different methods used in other departments to calculate and regulate workloads, but I couldn't convince the academic staff in the Computing Department to try them out. Believing in Logic to the extent that I did, I wasn't inclined to impose by force what I couldn't achieve by logical argument.

I was even less successful in getting advice from the College about how to calculate the amount and distribution of research grant overheads; and this was one of the areas where some of the most difficult problems arose in the Department. People couldn't agree whether research overheads should mainly support the groups doing the research or should support the Department as a whole. The College had no general policy about this, and different departments had widely different policies and practices. Discussions in our Department didn't produce any consensus either.

Although I tried hard to formulate general rules, I didn't succeed in convincing the Department. In addition, there were too many other problems that needed attention. These ranged from external problems of trying to get more resources from the College to internal problems of allocating scarce resources, such as office space, within the Department. I was surprised and disappointed to discover the extent to which people were unwilling to sacrifice their own personal interests for the greater good of the community as a whole.

I resigned as Head of Department, handing over to my successor in July 1999, and taking early retirement, at the age of 58, on 1st September 1999.

Professor Emeritus

Having left the Department, I planned to focus on writing my book about the application of Computational Logic to everyday life, aimed at a general, non-technical audience. But first there were a number of other matters that needed to be cleared out of the way, some academic and others purely domestic.

On the domestic side, I moved with my wife from our home in Wimbledon to a small hamlet in the West Sussex countryside. We extended the original seventeenth century cottage, added an oak, timber-framed summerhouse, and created a parking area. I did most of the planning and project management myself, and some of the timber framing and masonry. I enjoyed the change from academic work.

I also enjoyed the opportunity to combine academic work with extended visits to Japan, Australia, Portugal and Switzerland. These helped me to return to research and to recover from my period as Head of Department.

Writers' Workshops

Among other activities, I have been organising a number of Writers' Workshops on Logic and English, initially for PhD students at Imperial College, but more recently in Japan. The students present short, written abstracts of their work, and we discuss and debate how to improve the abstracts, by using concepts of clarity, simplicity and coherence derived from Computational Logic.

I have enjoyed these workshops more than my other teaching. Compared with my lecture courses, which were often a stale recitation of predetermined conclusions, the workshops have generally been an exciting, mutual learning experience. The students seem to enjoy them as much as I do. I get a chance to test my theories about the logical nature of human thought, and the students get to see how the theories apply to their own practical problems of communicating their thoughts more effectively to other people.

WHO and UNICEF

I had another opportunity to apply Computational Logic to practical problems, when Tony Burton, working at WHO in Geneva, contacted me in 2009.

Tony belonged to a WHO/UNICEF working group tasked with producing annual estimates of global, country by country, annual infant immunisation coverage. Since 2000, the group had been collecting immunisation data from national authorities, together with data from national surveys. The different kinds of data are often inconsistent, both independently and in combination. The group needs to reconcile inconsistencies and publish an independent estimate of the actual immunization coverage. These estimates are often controversial and may be disputed by both national authorities and technical experts.

Tony contacted me to see if I could help the group to formulate their informal rules and heuristics in more rigorous, logical terms, to make their decision making more transparent and more consistent. Computer implementation of the rules was not a major objective.

The group had been considering various possibilities for formalising the rules, including the use of both logic programs and production rules. We had many discussions about the differences and the relationships between the alternatives. Eventually, we agreed on a formulation of the rules in logic programming terms, which we then implemented in XSB Prolog.

The WHO/UNICEF working group has been using the Prolog program since 2010. In addition to helping ensure consistency, the program documents the argument for every estimate. Because the rules are transparent, the estimates can be challenged; and if someone puts forward a convincing counter-argument, the rules can be refined to produce better estimates both in the disputed case and more generally.

The Book: Computational Logic and Human Thinking – How to be Artificially Intelligent

Both the Writers' Workshops and the work with the WHO/UNICEF working group confirmed my conviction that Computational Logic can really help people to think and behave more intelligently. This helped to encourage me in my work on the book.

When I first put this story on my webpage in 2002, I had made enough progress to acknowledge that I was actually writing the book. But it was proving more difficult than I had expected to make the book accessible to a non-technical audience.

The book was finally completed and published in 2011. Although it was not intended as a textbook, it has been used as a text in at least six different universities, in both computing and philosophy departments.

The Meaning of Life

The book includes a chapter on the Meaning of Life. Admittedly, the title of the chapter was designed to attract attention, but one reviewer seemed to dismiss the title altogether by pointing out that the Life in question is that of a humble wood louse. I was disappointed because I had intended the wood louse as a metaphor for other agents more generally. In particular, I hoped that readers would notice that it is perfectly logical for an agent to have an intelligent designer, but also to have, as its over-arching goal in life, the goal of multiplying its genes by having as many children as possible.

I also hoped that the example would help to clarify one of the main claims of the book, that goals are more important than beliefs.

Goals and Beliefs

Most of my research has been associated with the field of logic programming. But I had begun to be concerned with integrity constraints and their relationship with logic programs, starting from around 1985, when I began working with Fariba Sadri on integrity checking for deductive databases. Integrity constraints also play a fundamental role in abductive logic programming (ALP), where they constrain the assumptions that can be made in solving problems. In ALP agents, integrity constraints represent goals, logic programs represent beliefs, and assumptions are actions, which, if they are successful, enable agents to satisfy their goals. The notion of intelligent agent in my 2011 book is an elaboration of such ALP agents.

ALP has been around since the late 1980s, and has had moderate success as a framework for "knowledge representation" and problem solving in artificial intelligence. It might have been more successful if its proponents could agree on its "semantics", in particular on the relationship between logic programs and integrity constraints.

In the context of ALP, there are two main, alternative views of this relationship. The theoremhood view regards abduction as generating assumptions that together with the logic program logically imply the integrity constraints as a theorem. The consistency (or model-generation) view, on the other hand, regards abduction as generating assumptions that together with the logic program make the integrity constraints true. In the theoremhood view, the logic program extended with the assumptions is a set of axioms (or theory). In the model-generation view, the extended logic program is a definition of a semantic structure (or model).

The theoremhood view has many attractions, not least of which is that it has provably complete proof procedures. I fell victim to its attractions myself, by contributing to the development of such a proof procedure with Tze Ho Fung. But as I argued earlier, completeness proofs can be a sign of weakness. More insidiously, because of their attractiveness, they can keep you from exploring better alternatives. It took me a long time to realise that, even if it lacks completeness proofs, the model-generation view is the better alternative.

The model-generation view of integrity satisfaction is especially appropriate when integrity constraints represent an agent's goals. In such a case, the agent's mission in life is to perform actions that, together with any external events that are beyond its control, generate a world that makes its goals true. The agent's beliefs determine how the agent views the world, creating abstractions from more concrete observations, composite events from more primitive events, and plans from more primitive actions. Understood in this way, an agent's goals are the driving force of its life, and its beliefs play only a supporting role.

Computing as Model Generation

Having completed the book, and having argued the case for understanding the goals, beliefs and actions of intelligent agents in ALP terms, I returned to more technical work with Fariba Sadri, investigating how to use ALP for practical computing.

Already in the mid-1990s, we argued that integrity constraints in ALP provide the functionality of production rules, active databases and BDI (Belief, Desire, Intention) agent programming languages. But our arguments seemed to have little impact. For one thing, we didn't have an efficient implementation. For another, there was a huge flaw in our approach, which made a practical implementation impossible: the frame problem.

The first part of the frame problem is how to represent in logical terms that most actions and other events change only a few facts about the state of the world, initiating some new facts, and terminating some old facts. All other facts, which are neither initiated nor terminated, simply persist by inertia. The second part of the problem is how to reason about such changing states of affairs. If you read the Wikipedia article on the frame problem (in June 2015), it seems that everyone believes that both parts of the problem were solved a long time ago, with the event calculus being one among many solutions.

In reality, the first part of the problem has been solved, but the second part has not. Almost all of the logical representations that solve the first part of the problem make it necessary to reason, in effect, that, whenever a collection of events occur at the same time, every fact that held before the occurrence and that is not terminated by the occurrence continues to hold after the occurrence. This is certainly not practical when the facts represent a moderately large database or large amounts of data manipulated by a computer program.

All practical database systems, programming languages and even BDI agent languages solve the second part of the frame problem by destructively updating states, deleting facts that are terminated by events, adding facts that are initiated, and leaving facts that are not terminated simply untouched. In effect, they solve the second part of the problem by ignoring the first part of the problem altogether.

Fariba and I have developed a variant of the ALP agent framework that aims to be a practical, but logical alternative to conventional computer languages. We call the framework LPS, because it was originally intended to be a Logical Production System-like language. Since then, the framework has been extended considerably, and I believe that it now has the potential to serve as a single, unifying framework for all areas of computing, including programming, databases and artificial intelligence "knowledge representation" and problem-solving.

LPS has the potential to be a practical framework, because it solves the second part of the frame problem in the manner of all other practical computer languages, by destructive updates. It also solves the first part of the frame problem by representing change of state in logical form, but without using the logical representation explicitly to perform change of state. Instead, the logical representation is an emergent property that is true in the sequence of states and events that results from destructive change of state.

Giving up the theoremhood view of goal satisfaction, and replacing it with the model-generation view, plays a key role in this solution of the second part of the frame problem. In the theoremhood view, destructive updates are not logically acceptable, because they are equivalent to changing the axioms during the course of trying to prove a theorem. In the model-generation view, destructive updates are acceptable, because they simply construct a model piecemeal. The model is constructed in the same way that the real world unfolds, existing at any given time only in its current state, and changing state by destroying the past. But in its totality, the real world is the complete collection of all its states and events, past, present and future.

There are many implementations of LPS, mostly the result of student projects. Unfortunately, none of them are as yet suitable for wider use.

Life in the Stone Age

I don’t work all of the time. The best time for me to work is in the morning, and then intermittently throughout the day. Some days I don’t work consciously at all.

Living as I now do in West Sussex, I don’t have to go far to immerse myself in the English countryside. The South Downs are not far away, and I can also walk straight out of my garden or across the road into the adjacent fields. My neighbour, who farms the fields, lets me wander over them with few constraints.

One day about eight years ago, I was walking in the field across the road when I noticed some worked flint lying on the ground. For several years, I had been looking for prehistoric flint artefacts off and on, mostly in the South Downs, where there are Neolithic flint mines. I soon discovered that closer to home, all around me there were the remains of prehistoric activity, mostly dating to the Mesolithic period about 8,000 years ago. Since my first discovery, I have identified three separate Mesolithic sites and collected a large number of flint artefacts, including microliths, arrowheads, scrappers and knives.

I often think, when I am fieldwalking, looking for signs of prehistoric life, how much our lives have changed over the millennia, how quickly they are changing today - and yet, how much they have remained the same.

Search for Truth

Looking back at my academic work, I like to think that most of it has been driven by the search for truth, with Logic leading the way.

The search began in high school, triggered by my extracurricular reading of such books as Joad's "Guide to Philosophy". When I read about Plato's philosophy of ideas, I was convinced that it was true. And when I read about Aristotle's empiricism, I was convinced again, but this time that a contrary philosophy was true. It couldn't be that both philosophies were true. But Joad offered no guidance to distinguish between them.

The first year mathematics course at the University of Chicago introduced me to mathematical logic, which seemed almost magical in its symbolic form. Mathematical logic seemed to be able to create truth out of nothing. I decided to major in mathematics at the University of Bridgeport, partly because mathematics is the language of mathematical logic, and partly because it seemed to show that indisputable truth is possible. I hoped it might help me to find other truths elsewhere.

My search continued at Stanford and the University of Warsaw. But I began to doubt that mathematics would help to solve such life and death problems as the war in Vietnam, which troubled me at the time. I never questioned the relevance of Logic, because to my mind the logic of common-sense left no doubt that the war was wrong. But I questioned the purpose of mathematical logic, because it seemed to me that it had become a branch of pure mathematics, and that it had lost touch with the original purpose of Logic, to help people think more clearly and more effectively.

Ideally, I would have continued my studies of Logic in a philosophy department. But I didn't have the necessary academic background. I found myself doing a PhD in a computer science department at the University of Edinburgh instead. Fortunately, the PhD, which was about using symbolic logic to mechanically prove mathematical theorems, didn't require any knowledge of conventional computing.

The topic of my PhD was not one I chose for myself. Nor was it on a direct path to my ultimate goal. But it gave me an entry into the field of artificial intelligence, where I worked on the development of logical methods that could be implemented by means of computers. Although I had little enthusiasm for the goals of artificial intelligence, I learned that the same logical methods I was developing to prove mathematical theorems, could also be used for other, less mathematical kinds of problem solving. I was encouraged by the thought that the same logical methods, used to make computers more intelligent, could also be used by people to improve their own human intelligence.

My work has also benefited from attacks against logic by other researchers working in artificial intelligence. These attacks drew attention to weaknesses in my theories and helped me to identify where the theories needed to be improved.

Perhaps the biggest weakness of traditional mathematical and philosophical logics is that they focus on disembodied, pure thought. Even logics, like the event calculus, that are concerned with actions, events and changing states of affairs just deal with thinking about change, without actually performing it. I believe that this is another problem that the model-generation semantics solves: Given a current state of the world and a stream of world-transforming events, the task of an intelligent agent is to perform actions that, together with other events occurring at the same time, change the state of the world, in order to make the agent's goals true in the resulting totality of states and events, as viewed in terms of the agent's beliefs.

For me, the model-generation semantics reconciles the Platonic-like ideals of an agent’s internal goals and beliefs with the Aristotelian-like empiricism of external states and events. It reconciles declarative representations, which are grounded in the definition of truth for sentences in logical form, with imperative representations, which require actions to make declarative sentences true.

My search for truth has come a long way. It started by looking to prove theorems as logical consequences of axioms; and it continues. But now the truth, I believe, is that the life of an intelligent agent is to act in such a way that the goals it seeks to make true are made true as best as possible. The hope is that the goals of all agents, taken together, are at least mutually compatible.