# Structures, systems—and people

**John N. Crossley**

Monash University, Australia

**Abstract**

In 1982 I was Chairman of the Mathematics Department at Monash University. Not long afterwards I became Chairman of the Department of Computer Science, both in the Faculty of Science. In 1991 the Computer Science Department became part of the Faculty of Information Technology.[1] (At Monash we also have a Department of Electrical and Computer Systems Engineering in the Faculty of Engineering.)

These different contexts have made me reflect on how the various groups of people approach research and teaching. Some items are held in common: structures and systems are central concerns for mathematics and information technology, but not necessarily for other academic disciplines. People's attitudes are different, but dialogue opens up new possibilities.

I would be pleased to share what I have learnt in these changing environments.

[1] A history of the situation, in what was to become the greater Monash University, leading up to the foundation of the faculty may be found in Sarah Rood, *From Ferranti to Faculty: Information Technology at Monash University, 1960 to 1990*, Clayton: Monash University Custom Publishing, 2008. Sarah Rood tells the story of the foundation of the first faculty dedicated to computing and information technology in Australia. Covering the period from the late 1950s until 1990, when the Faculty of Computing and Information Technology at Monash University was formed, it spans the establishment phase of the discipline of computing.

Some years ago a very distinguished German professor said to me, when I was discussing the new ways in which universities were changing: "Universities change very slowly, but the rules change every week." I expect that you are feeling the latter rather sharply at Sophia.

Monash University was founded in the early 1960s, the time of the founding of the Mathematics department here at Sophia. It was the first of the new universities in Australia and its creation followed a wave of new universities in the United Kingdom. It was supposed to be predominantly science and engineering but in fact it became a traditional university with a strong Arts faculty. Like British universities it had a number of faculties—the number has varied over time and I will talk about that later. Mathematics was in the Science faculty; there were neither computer science nor computer systems departments anywhere. In 1968 the university decided to start a department for computing. Its name was originally "information science" but this was changed with the appointment of the first professor. His name was Christopher Wallace and he had a background in Physics but no formal training in computing. In fact he was a polymath with an excellent knowledge of statistics and a very wide range of interests in science generally. He is chiefly remembered for the Wallace multiplier, a circuit design used in all computers today.[2] The embedded algorithm is much faster than anything that preceded it.

At his insistence the name of the department he was to head was changed to "Computer Science" and it was decided to put this department in the Faculty of Science. The department quickly attracted a large number of students and caused terror in the hearts of the established departments of physics and chemistry. So much so that quotas were imposed on the number of students the Department of Computer Science could take in its various subjects. Further, it was not classed as a laboratory discipline so it was not granted extra staff and money for laboratories

[2] Christopher Stewart Wallace, Suggested Design for a Very Fast Multiplier. Technical Report UIUCDCS-R-63-133, Department of Computer Science, University of Illinois at Urbana-Champaign, 1963, Correlated round-off errors in digital integrating differential analyzers. *Computer Journal* vol. 7, pp. 131 – 134, July 1964 and A suggestion for a fast multiplier. *IEEE Transactions on Electronic Computers*, 14 – 17, February 1964, reprinted in E.E. Swartzlander, *Computer Arithmetic*, Vol. 1, IEEE Computer Society Press Tutorial, Los Alamitos, CA, USA, 1990.

but had to make do with a lower allocation—just like Mathematics. The result was that the staff were overworked and facilities were very limited. The same was true of mathematics.

Unfortunately this shared burden did not lead to close relations between mathematics and computer science. I think this was largely due to two causes. One was that mathematicians felt so oppressed that they wanted all the resources they could get, and they did not wish to further antagonize the other, laboratory-based, departments in the faculty of science. The other was that there was a very poor understanding of what computer science was. This lack of understanding seems to come from two sources: on the one hand I think it is still not clear what computer science is, even to computer scientists, and on the other everyone was starting to use computers in their work and therefore felt they knew sufficient about computing and there was no need for a special department. This lack of understanding continued for many years—and still does, though to a lesser extent.

Let me digress a little. It may be said of me that I never did computer science, or that I always did computer science, for my doctorate is in mathematical logic and more specifically what is known as the theory of computability. This was the area that led Alan Turing into inventing the theoretical stored-programme computer, and then to the design of the first British computer built at the National Physical Laboratory in the late 1940s. However, when I began to study for my thesis an applied mathematician said to me, “Why don’t you do *real* mathematics?” He did not consider logic “real” mathematics. On the other hand, Michael Atiyah, a very distinguished analyst and topologist said to me just a couple of years later that he thought mathematical logic was “difficult”. Nevertheless, in general, the mathematicians around me in Oxford shared the applied mathematician’s view.

The Monash Computer Science Department remained beset by problems. Some of these were aired in the Professorial Board, the principal academic committee of the university, and in 1982 an enquiry was set up into the status and needs of computer science. The enquiry was headed by the two deputy presidents of the university, so it was clear there was concern at the highest level. I was asked to serve on that committee as a representative of mathematics with some interest in computing. In the middle of the deliberations of the enquiry the chairman of

Computer Science was hospitalized and had a quintuple bypass. He was replaced but the department was losing staff rapidly because of the uncomfortable position for its staff. The new chairman quickly left. At the time I had recently become chairman of the Mathematics Department, but then the enquiry asked me to head the Computer Science Department. I knew some of the people, and a little about computability, but nothing about circuit design or databases, for example. As people say today, I was on a steep learning curve. I had one other major factor in my favour: the teaching staff of Computer Science were very appreciative of my interest and concern.

To me this is the most important thing in work: dialogue and understanding between people.

Because many people in the university had taken up some form of computing the university developed a support service that centred on the Computer Centre. This was run by an engineer who regarded his brief as saying that he should provide the greatest good for the greatest number. One consequence of this was an excellent provision of basic services and an almost compete neglect of specialist services such as high performance computing for chemists, engineers, fluid dynamicists and so on. On the academic side the faculty of engineering the department of Electrical Engineering had developed an interest in computer systems. This meant that serious academic interests in computing were now evident in two separate faculties. It also led to some tensions between Computer Systems Engineering and Computer Science, tensions that have seem not to have largely, though not entirely overcome.

In the course of the enquiry that I have mentioned it was suggested that Computer Systems Engineering and Computer Science should get together formally. To me this seemed premature. Instead I suggested that there be a party for members of the two departments. In the end there was a short meeting *and* a party. Some dialogue emerged, but largely the members of the two departments talked among themselves. Getting people to understand each other takes time and patience. More recently I have been involved in several attempts to develop cooperation between members of our new Faculty of Information Technology, about which I shall say more later, but which does not include mathematics. Interest came primarily from Medicine and over

several years various meetings were convened to look at possible projects. For a long time these were characterized by the Medical people saying: “We have a problem”, and the Information Technology people saying, “We have solutions”. Unfortunately neither fitted the other.

Let me make a small aside: the same situation had happened before when Mathematics, which, in the British style, includes statistics, offered its services to the biological and other departments.

I am pleased to say that in the last two or three years cooperation has developed and there are research groups now involving people from Medicine and Information Technology working together very effectively.

Let me now try to analyze what led to good cooperation.

In my own case, when I became chairman of computer science, I started looking at the research problems being investigated in the department. People also came and asked me questions of a mathematical nature. Often these were statistical, a subject about which I know virtually nothing, but I did know the appropriate people in the mathematics department and that encouraged some dialogues. For myself, I discovered that much computer science was taking ideas from mathematics and developing them in a more practical way. I do note want to stress this too much, but the questions of most interest to me involved the development of languages that could be used on a computer but had their roots in traditional mathematical logic. Chief among these languages were variants of PROLOG, a language that has developed considerably over some thirty years. As a logician my interest in looking at formal languages centred on their ability to give us an understanding of mathematical structures. I quickly came to see that a computer scientist was not interested in being able to capture such structures *in principal* but wanted to get results in “reasonable” time. This seems rather like the difference between having a theorem that a certain differential equation has a unique solution and asking for that solution—the difference between these last two sometimes is extremely large, perhaps unbridgeable. Happily, because of the way that mathematical logic has developed, the gap for formal languages is not as great as that for solvers of differential equations. One of the effects of this large gap in

the case of differential equations is that it has led to the development of a whole new field, that of numerical analysis.

Here again, I have seen great changes. When I was at Oxford, first as an undergraduate but later teaching there, a new chair was founded, in Numerical Analysis, and by chance the new professor had been to the same grade school as I had. He had a very hard time persuading the mathematicians around him that he was involved in a reasonable pursuit. Many felt that he was not doing “real” mathematics. Nowadays I think it would be taken for granted that if one wants accurate numerical solutions to, say, differential equations or to some problems involving large matrices, then an understanding of the associated numerical analysis is essential. Moreover, the skills and intelligence required are great, though they are often very different from those required to prove existence theorems for solutions of differential equations.

What I think these two examples reveal is that, although traditionally mathematicians have been highly educated in the study of particular structures, there remain many areas that are not only new, but have not been considered previously: areas that are essential when we are using new technology. Of course there are other areas that emerge and may simply be fun. The prime example here is surely the study of fractals, an area that allowed the generation of pretty pictures using high-powered computing machines. Nowadays such fractals also have practical uses in image processing.

In order to understand these areas I believe that a thorough training in dealing with systems and structures is required. A couple of years I went to a seminar given by an artist and a mathematician. In the course of the seminar, which was about artificial life and the generation of artistic pictures, usually of landscapes in this instance, the speakers mentioned that their most recent work was with the scientist (Ian Wilmut) responsible for Dolly the sheep—the first cloned animal. What was startling then was the fact that the biologically trained scientist did not know about systems!

Anyone in a computing or mathematics department knows about systems, and also about the structures on which the systems work. I would find it hard to be a mathematician if one did

not know about some sort of structure and systems. Group theory is a prime example; communication networks are another. To me, one of the things that makes the structures of computer science more interesting, and often more difficult, is that there is a new variable, time. The most obvious manifestation of this is the simple difference between

$x = 5$ and $x := 5$

In the former there is a static inequality, the value of the variable is 5, in the latter the variable is set to 5, whatever it was before. Studying group theory the groups are static entities, in networks the traffic flowing through them is the prime concern, though connectivity is also important.

(On one of my early visits to the Philippines around the end of the seventies I attended a seminar on graph theory given by Koh Khee Meng from Singapore on graph theory. At that time graph theory was relatively new and viewed with suspicion by some mathematicians. However, some of the work was employed directly in the layout of transmission lines for electricity.)

These examples, I hope, show how one may be working on the same situation, but the approach will be different, perhaps very different indeed, for the mathematician and the computer scientist. Often there will be problems that neither can solve, but there are many occasions when sharing knowledge will lead to a solution.

There is a danger here, for mathematicians and engineers have long had an equivocal relationship. All engineers are forced to do what often seems to them a great amount of mathematics, and which also often seems to be of dubious usefulness. The same was true of computer scientists when I was first involved. The standard course required a certain minimum of mathematics, mathematics which, for the mathematics students, was pretty easy: calculus and linear algebra. The engineering students could not see the relevance, or at least *did not* see the relevance until a year or two later. Eventually they came to realize how important the mathematics was. In my own experience computer science students are very similar. However, they are from a new breed, which has grown up using technology before they understood it—if they ever did!

In moving into the Computer Science Department I experienced this new kind of student. I also found that they learnt things differently from mathematics students. The mathematics that I learnt as a student I still remember, though I am not sure I could deal with Weierstrass p-functions very well without looking them up! The knowledge I acquired fifty years ago is still in my memory, even if a little buried. I found computer science students complained: “We did that two years ago. I have forgotten it all now.” The knowledge that people in computing use is constantly changing and the space in human memory seems to keep getting refilled—and the old material lost. I found this quite disconcerting, but there was compensation: the computer science students learnt new material much faster than the mathematics students, in general.

There was also a different in approach to teaching and learning. Not only did they question why they had to learn certain mathematics, they were very vocal about it. There was a big difference between the treatment of students in the two departments. In both there were student-staff liaison committees. In the Mathematics Department these committees did practically nothing, it was even hard to find student representatives. In the Department of Computer Science there were numerous student representatives, and they spoke up fearlessly in the liaison meetings we had. On first encounter this was very surprising. However it did not take long to realize that the students were very serious minded. Indeed I think they take their responsibilities much more seriously than the mathematics students ever did. A dialogue existed between the students and the staff. As a result lectures and teaching generally improved. The students also developed not only a better understanding of computer science but also a better understanding of their faculty members.

Once again, the importance of dialogue is evident.

Sometimes it is difficult to establish a dialogue and I think this is often the case when the interlocutors come from departments that are very close. Similarity and contiguity, says Freud,

breed distrust, rivalry, comparison, even, perhaps, self-hatred or self-doubt projected upon the nearby other.[3]

This has certainly been the case for the Department of Computer Science, originally in the science faculty, and the Department of Electrical and Computer Systems Engineering. In the case of Sophia, from looking at the interests of your combined department, I feel this may not cause such great difficulty here. However, there is another problem, I think it will be quite difficult for some members to understand what the other person is doing and why.

Again I turn to my own experience. When I was an undergraduate I was in an Oxford college. In all years, there was a total of something over a dozen of us doing mathematics in a college of more than 200. My immediate neighbours, who became my friends, were doing a wide range of studies: Classics, History, English, Modern Languages, Chemistry, Biology. So, over the course of time, but a very short time, I learnt how to speak to these people and to get them to speak to me about the work they were doing. Perhaps this is the most valuable thing I learnt in Oxford!

Let me return to the cooperation between the Faculty of Information Technology and other faculties. As I have mentioned, projects have developed with people in Medicine. One of the most highly developed has involved analysis of DNA sequences. The number of such sequences is huge and in order to make a sensible attack on matching given sequences one needs both algorithms (from computer science) and biological information (from the biochemists). The best algorithms are informed by the experiments of the biochemists. Having an effective dialogue between the two faculties means that questions can be formulated *in a common language*. So it is no longer the case of "We have problems" and "We have solutions" but rather, "We are working together to solve some problems".

---

[3] Marjorie Garber, *Academic Instincts,* Princeton: Princeton University Press, 2001, p. 55. See her note 2 for the original Freud.

In my own case, my research has been profoundly affected by dialogues with my colleagues, yet my most recent research in logic has been using techniques that are absolutely standard in the (pure) mathematical logic community; it is just that they are being employed in a different way and can be put to much more practical use.

Now let me return to the history of my university and to a much more difficult problem than moving from a mathematics department to a computer science one. I am talking about the formation and development of our new Faculty of Information Technology.

New faculties, new departments, indeed amalgamations in general are often driven by economic concerns. The Faculty of Arts in my university has suffered a succession of cuts, driven by economic imperatives. In recent years some departments have been abolished, for example, Classics, or amalgamated, for example, Modern Languages, which used to comprise departments of French, German and Spanish. That was the first big change. In the last year there have been further "rationalizations". History and Philosophy, the two strongest departments in the faculty, have been transformed into the School of Philosophical, Historical and International Studies. (This now has the acronym SOPHIS, which I am tempted to read as "Sophist" rather than "Sophia".) I do not expect these economy driven changes to cease.

The case of the formation of the Faculty of Information Technology was also driven by economic concerns, but also by other concerns and by a change in government policy. The particular matters are not relevant here. The mechanism was that larger universities were being formed from various tertiary institutions. My university absorbed a teachers' training college and an institute of technology, and other bodies. It therefore became much larger and this provided an opportunity for some reorganization. Indeed our Faculty of Information Technology now operates in six of the eight campuses that we have.

I have already mentioned the difficulties that Computer Science laboured under in the faculty of science, the Department of Information Systems, meaning "business information systems", also languished, but it was then in the Faculty of Business and Economics. The university enlargement provided an opportunity to form a new faculty devoted to matters related

to computing and information technology. However the departments involved were numerous and of very distinct characters. On the more academic side was computer science, which had strong mathematical requirements; on the business side was the Department of Information Systems; and then, there were a number of rather smaller technology oriented departments from the Caulfield Institute of Technology. Not only were these very practically oriented, they taught in a different way. I would say that they aspired to teach competence rather than understanding. In addition, research had been discouraged in those departments. An interim dean was chosen: the director of the Computer Centre, whom I mentioned earlier. He had a large influence and I am sorry to say that in my view he held back the development of research by at least five years.

The departments from the institute of technology had developed certain work practices that were very different from those in the original university. Nevertheless there was a great deal of good will for everyone felt that, with the establishment of the new faculty, there would be a brighter future. First, however, we had to learn to work together. Again this was a question of developing dialogues. Unfortunately the interim dean discouraged this. I would say that his policy was to keep the departments isolated from each other so that they did not cooperate trouble him with the problems they had because resources were still limited. This policy was aided by the fact that the institute of technology departments were some ten kilometres distant from the Department of Computer Science. This distance also discouraged dialogue and certainly created an “us and them” mentality. However, we eventually acquired a new dean and he encouraged the various departments to get together. Earlier I spoke about having a party to get people together and our new dean called a “retreat” for people from the “old” university and the institute of technology. We went away to a seaside hotel. The first night people from the “old” university sat at some tables and the people from the institute of technology sat at other, different ones. The second night we were all happily mingling together.

Our new dean had arrived in the time of the IT boom, so resources came available and there was no longer any need to fight over money, there was enough for all. Slowly, right across the faculty, people were encouraged to talk to people from other departments; joint projects and joint seminars were developed. Research teams were formed that bridged the original

departments. Nevertheless, to a very large extent the departments continued to teach as they had done in the past. It was not until a dozen or so years from the founding of the faculty that the latest, that is to say the third, dean seriously took in hand the question of forming an integrated faculty. Dialogue has been encouraged, indeed required. The initial prejudices, which had been nurtured by the first dean, still persisted. This has been the case more particularly among the older members of the faculty. The IT boom burst, not only was there no money to develop new projects, we had to have reductions in our faculty numbers. Nevertheless, we did acquire a certain number of younger people who were ignorant of the previous history. They have conversed with people from all over the faculty. Slowly we are developing a sense of unity.

Let me now try to analyze some of the above and to raise questions that, I believe, may be relevant to Sophia.

When I looked at your web site I noted that the following were listed as areas of interest:

Algebra (especially group representations), Analysis, Geometry, Statistics, Biomedical Mathematics (to me this subject has at least two meanings: biomathematics and statistics, but now also embraces string matching), Manufacturing Systems Engineering, Information Systems, Library Science, Neuroscience, Pharmacology, Computer Networks, Databases, Educational Technology, Perception & Robotics.

At Monash, virtually all of these areas fall within either mathematics or information technology—the vast majority in Information Technology, but we would also add a number of areas such as data mining, business systems.

To the insider these seem to form a huge range of disparate areas; to the outsider they all look very technical and mathematical.

What divides these areas, and what unites them? In all of the areas that I have listed one is concerned with **a**bstract structures, which underlie what we are studying. Of course the types vary: some are straightforwardly mathematical, such as groups and geometrical objects. Others may be networks, whose underlying structures are graphs; classificatory, where, as in library science and information systems, one needs not only databases but ways of dealing with them

and studying them, plus techniques for extracting data from them, and further there are social systems if, like many people in my faculty, you are studying the impact of information technology on society.

It is interesting to note that systems, which are so important to all in mathematics and computing, are not necessarily studied as such in other disciplines. Of course, individual systems are studied, but this can be done without an awareness of what a system is.[4] This was brought home to me listening to a seminar by a computer scientist, Mark d'Inverno, and an art professor, Jane Prophet, who described some of their work with Ian Wilmut who was one of the people responsible for Dolly the sheep. He had not thought of systems, apparently! Not all systems are the same, as we well know, but it was still a surprise to me to be introduced to operating systems. I was familiar with algorithms, but the basic property of algorithms, for me, is that they terminate. With operating systems it is important that the algorithm does *not* terminate, otherwise the system crashes.

How are the areas studied? When I was a student it was standard for individuals to work on a problem on their own. If you were a graduate student you had a supervisor. No-one else seemed necessary. Things have changed enormously. Even in order to study some very narrow abstract mathematical subjects, such as number theory, one needs to know a great deal of algebra and analysis—or at least to have access to people with such knowledge. In computer systems and networks the need for different knowledge bases is even more evident. In recent work I have done none of the three of us in the group could have done the work on our own; none of us had the technical expertise to fashion all the pieces.[5]

---

[4] Indeed the abstract study of abstract systems is not well represented in the literature. The principal reference I have is quite old: Ludwig von Bertalanffy, *General systems theory: Foundations, Development, Applications*, New York: George Braziller, 1968.

[5] Iman Hafiz Poernomo, John Newsome Crossley and Martin Wirsing, *Adapting proofs-as-programs: The Curry-Howard Protocol*, Springer Monographs in Computer Science, Springer, New York, 2005.

Sometimes there is a very substantial difference in approach. Researching the higher domains of abstract mathematics one can often, and easily, forget about any involvement with the "real" world; working on operating systems one cannot. I found it inspiring to become a little more closely, but not too closely, involved with the "real" world when I moved into the Computer Science department and learnt about the programming language PROLOG. I have also found that between very narrow disciplines there can be huge voids just crying out for exploration—and very quickly yielding fascinating results.

How are the different areas taught? Because of the very different nature of the subject matters and also the different approaches used by the various professors and departments, we have worked hard to prepare a quite detailed syllabus for every individual subject. The details go as far as specifying what is to be taught in each lecture. This does not remove differences in presentation, but it does ensure that students who have done a course anywhere in the faculty—and please remember that our Faculty of Information Technology is split over six campuses—can go on to succeeding courses at any campus. I believe this cooperation in teaching has also led to better understanding between the various faculty members, and let me not forget that the abilities of these faculty members are quite varied. Some have very little mathematics, some are quite expert; some are adepts with machines; others find them a necessary evil.

The changes in teaching have been reflected in our attitudes to research. First of all there has developed a certain amount of cooperation—limited I admit—across the campuses. There has also been a change in our understanding of our role.

Let me compare the situation of mathematics. When I was a student, the number of graphs you would find in a newspaper was quite small; nowadays you will find them in every newspaper, especially in the business pages. Mathematicians in my country have protested about the decline in mathematics in universities, by which they mean that fewer faculty members are employed now than in former years.[6] However, surveys show that the general level of mathematical knowledge in the general population has steadily increased. In my own specialty,

[6] I admit that the number of students has gone up!

mathematical logic, whole research teams have vanished, but every computer science course teaches some logic; even electrical engineers know about Boolean logic and circuits. So we have seen a decline in the number of specialist mathematicians but an increase in the number of people using mathematics, and their level of ability.

Our view of our role in doing computer science, or indeed any computing, has changed from one of looking at problems *within* our discipline, to looking at problems to which we can bring our knowledge and distinctive point of view to bear on areas that have not been considered in such ways before. The new insights we bring are much appreciated. This does not make research any easier, but it can make it more interesting. It also requires the establishment, not simply of a dialogue, but rather of a common language—a language that incorporates terms and understanding from both disciplines—or even more than two disciplines in some cases.

So what are the lessons we have learnt at Monash?

The first, and last, lesson is that talking to each other is the first requisite. There can be no understanding without dialogue, and that dialogue has to be developed in a common language. Of course, there has to be the will, the desire or, unfortunately sometimes, the pressure, to start a dialogue. We found that quite difficult in our new Faculty of Information Technology. To some extent this was a matter of distance: the two main campuses are about ten kilometres apart and initially there was no easy public transport. More important, however, was the background that people had: the differences in styles, and approaches. This applies to both research and teaching. Even the students were different. As I noted above, computer science students wanted to know *why* they should study certain theoretical matters (especially mathematics) at a given time; in the 1980s and 90s, in the Mathematics Department it was always assumed—by both students and faculty—that the professors knew how things should be ordered and taught. That attitude has changed. This has been partly because the student numbers fell steadily in mathematics. Considering the explicit needs and desires of undergraduates has led to an increase in numbers—and significant changes in the syllabus. One measure of the change is that the new head of

mathematics was in the Information Technology faculty for a long time, though it is true that she was originally a mathematics graduate.

One painful thing to learn has also been the very practical need to recruit enough students to pay the salaries of the faculty and the support staff. One can have high ideals and excellent standards, but if there are no students the people who guarantee those standards will leave the university to find posts elsewhere if they cannot be paid. Our previous president was very conscious of that, and although his aim was to make sure that Monash was very definitely a world-class university ion research, he was also very conscious that unless there was sufficient income to the university that would not be possible. So he made compromises that some of us did not like. One thing I have noted all through my career is that people in a given discipline generally agree that there is a central core of knowledge that must be taught. On the other hand the slightest probing reveals that there is virtually no agreement on what that core is. To give a very specific example: in the Department of Computer Science we were trying to reduce the number of subjects taught. There was a strong push to have one particular course taught in the penultimate undergraduate year. I was chairman at the time, so I asked who would teach the course. Naturally everyone already had at least as much work as they could handle. No-one volunteered. So the course was not taught. Pragmatism is important at times.

In integrating our new Faculty of Information Technology we have developed a first year course with most students taking the same subjects. This has been quite difficult for two reasons: one is the variety of subject matter that needs to be covered these days; the other is the question of how much mathematics and computer programming should be included *for everyone*. In order to achieve this, nominations were called for a committee. This committee ultimately contained representatives from across the faculty, many of them quite junior. The compromises that were reached initially satisfied no-one, but over just a few years cooperation and discussion have led to a stable first year. The effort demanded, and the good will required, have been very large. The result is that, after their first year, students are much better equipped to make a sensible choice of subjects in their later years: the whole breadth of the faculty is open to them. Moreover, everyone

has a basic level of mathematics, and this is, in my view, essential for anyone who is studying a structured subject, such as practically all mathematics and computing disciplines require.

One question we have not resolved is how to look after the very best students. Over the last twenty years or so student numbers have increased very rapidly and universities in Australia are now close to their US counterparts in that we take close to 50% of the population cohort—that is to say, the young people in their late teens. The top 5% of the population are as good as they always were, but the "tail" is much longer. These top 5% now form only 10% of the student population whereas a few decades ago they formed something like 50%. There are many reasons why we have failed to treat them "properly", which in my view means teaching them more advanced material than the average. One is that "élitism" has been frowned upon—though we still talk about "élite athletes" and have a special institute for them in Canberra, our capital city. Another is that teaching and administrative loads have increased significantly so that we find it very hard to put on extra courses. Happily these students tend to be self-motivated and go and work on material on their own, simply briefly consulting their professors as needed.

In a recent discussion with the head of my department he expressed one regret: that mathematics was not part of our faculty. At the time of the formation of the faculty, back in 1991, that was not even contemplated; now mathematics is much closer to information technology in my university and, if the faculty was being formed now, it would be very sensible to include mathematics. All would benefit.

*****

I hope that from all of this you may find the right questions to ask yourselves here at Sophia. I do not expect that the answers will all be the same as for Monash, but I hope you will develop your own answers and I wish you all the very best for the future of your department and your university.