UNDERGRADUATE EDUCATION IN COMPUTING SCIENCE - SOME IMMEDIATE PROBLEMS

J. Tartar and J.P. Penny
Department of Computing Science
University of Alberta, Edmonton, Alberta, Canada

1. INTRODUCTION

In September 1970, Wegner [1] suggested that "computer science may become the numerically largest field of undergraduate education within the next decade." At the University of Alberta this situation may not be far off. The Department of Computing Science has 200 Honors and 300 Specialization (four-year professional program) students, more than in any other Department in the Faculty of Science, together with several hundred three-year general program students majoring in computing science. It may be, as Wegner suggested, that computer science can become a "central scientific discipline", perhaps taking over the role now played by pure mathematics. However, Wegner himself pointed out that the discipline has yet to change and mature before this stage is reached.

The combination of a large number of students and a rapidly developing discipline imposes considerable responsibilities on those charged with providing curricula in computer or computing science. Rather than philosophize about the nature of the discipline itself in this paper, we intend to discuss at a pragmatic level the problems we see at the present time in our own case, and the steps we are taking to resolve them.

2. BACKGROUND

The problems as we see them at the U. of A. arise from two sources: rapid growth in terms of students and staff, and the lack of a well established academic identity. It is apparent in our case that these problems have compounded each other.

The Department of Computing Science was established at the University of Alberta in 1964. The Department had its most rapid growth between 1966 and 1970, and now this period of growth appears to have ended. During the same period there was a corresponding growth of the faculty with its members being chosen without any attempt at emphasis on a particular area. The result was that a wide range of programs and courses were developed. A recent survey [2] suggests that the Department's program may be the broadest of any University in Canada.

This broad range has its drawbacks, among them the difficulty of achieving real depth and reputation in any particular area. Another result is a continuing debate between, for example, those who feel that education in computing science should largely be mathematics-oriented, and those who would prefer to see a more liberal approach. If one accepts the views of Amarel [3], the field of computing science itself is very wide ranging. Despite the amorphous character, we believe that at this stage of development of the discipline, a large department cannot afford to restrict its range arbitrarily.

Together with the stabilization of student numbers, the programs themselves have achieved a certain stability - though, given that there are competing views, it would be truer to say that a state of equilibrium has been reached. Given the delay between initial registration and graduation, there remains a year or so in which the number of students graduating will increase rapidly. We predict that we could graduate perhaps forty honors, fifty specialization, and one hundred general-program students each year for the next several years. A feeling of responsibility for this large a number of students obliges us to reassess continually both our programs and our basic objectives.

3. PROGRAMS IN COMPUTING SCIENCE

The problems of curricula development are twofold. One problem is that of designing a four year program in which some students can be prepared for direct employment and others for graduate work. The second is that of designing a program with sufficient flexibility to be both relevant and accessible to a wide range of students, including those not concentrating in computing science.

The Department offers 66 courses, of which 42 can be taken by an undergraduate. (Here a course involves 42 lecture hours). The undergraduate program includes courses roughly equivalent to most courses in Curriculum 68 [4], those reflecting special faculty interests, and a variety of service courses.

The U. of A. offers two four-year programs, Honors and Specialization, leading to the B.Sc. degree. For a four year degree, 40 courses are required, and the standard program requires that there be at least:

In Computing Science:
16 courses (Honors);
12 courses (Specialization)
In Mathematics*:
10 courses (Honors);
8 courses (Specialization)
In the Faculty of Science:
4 courses (Honors);
6 courses (Specialization)
In the Faculty of Arts:
6 courses (Honors);
6 courses (Specialization)

(*Courses in numerical mathematics and statistics are included in the Computing Science course offerings).

However, a fundamental change in the faculty's attitude towards these programs is occurring. We shall discuss the implications of this change in attitude more fully in Section 4.

For some time there has been general agreement that the computing science courses taken by a student should include a set of courses taken by all students (the computing science "core" program) and courses which gives concentration in one particular area of computing science. We are convinced that, if we asked a randomly selected group of n computing scientists to suggest a core program, we would have n different proposals. However, there is agreement in our case that the core program should include at least the following courses:

> In Computing Science: Introduction to Computing, Computers and Programming, Introduction to Discrete Structures, Data Structures, Numerical Calculus, Probability, Probability and Statistics, and Switching Theory.
>
> In Mathematics: Introductory Calculus, Algebra and Geometry, Calculus, Linear Algebra.

In the above, we have given the course titles used in Curriculum 68, rather than our own titles, where the courses are sufficiently similar.

Beyond this point, it has been difficult to reconcile the interests (or biases) of faculty in different fields. However, to fulfill the objective that students concentrate their computing science options in one field, we have divided the program into three streams as shown in Figure 1. These are Stream A, Numerical Mathematics; Stream B, Systems and Languages; and Stream C, Formal and Adaptive Systems. One result has been that, where there are to be additional required courses in the core, there can be argument that there be a balance between the different fields. In addition to the courses above, the following courses are required for Honors students:

From Stream A: Numerical Analysis I
From Stream B: Programming Languages or Systems Programming
From Stream C: Introduction to Adaptive Systems.

Division of the program into streams appears to us unavoidable, however much one may object on grounds of arbitrariness, or possible rigidity of the program. As well as encouraging some concentration of options, and balancing different interests in specifying the core program, there is the advantage that each course designed has to be part of a reasonably coherent subprogram. A similar expedient has been adapted at Rutgers [3], probably for similar reasons. In their case, the streams or "partitions" are identified as Hardware Systems, Software Systems, Numerical Applications, and Nonnumerical Applications and no doubt reflect a corresponding division of Faculty interests.

4. OPTIONS OUTSIDE THE DEPARTMENT

Over the last year or so, the prospect of computing science being studied by very large numbers of students has caused a general but fundamental change in attitude. A belief has grown that we should concentrate less on educating specialists, and encourage students who are primarily computing scientists to study coherent programs of courses in other disciplines, usually application areas.

Obviously, there has not been an overnight change. Two years ago, the Department persuaded the Faculty of Science to accept specified courses in other faculties - notably Business Administration and Engineering - in lieu of Science options. The options in Business Administration have been particularly popular with specialization students.

One year ago, the statement for the Honors program was amended to allow an Honors student to build up strength in an area related to Computing Science. A coherent program of eight to ten courses in a relevant discipline may be accepted in lieu of the Science options, and the requirement for courses in Computing Science and Mathematics may be reduced by two in each case.

The most recent development has been the establishment this year of joint working groups, which now exist in collaboration with Mathemtics, Engineering, Business Administration, and Medicine. These working groups, which we hope will remain in existence and possibly be added to, have these objectives:

1) To make clearer what is acceptable as a "coherent program" of courses in a discipline other than computing science.
2) To investigate the possibility of joint degrees.

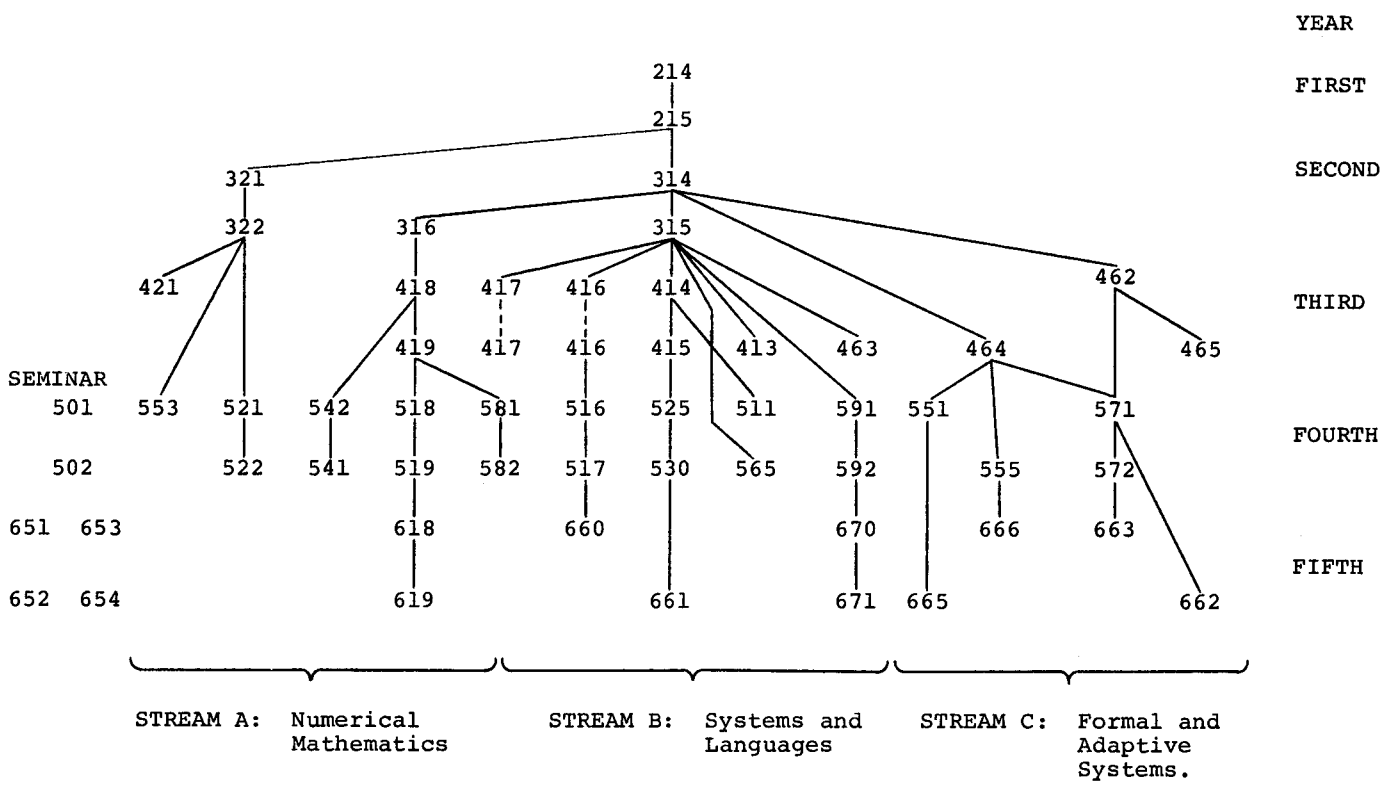

Figure 1. Computing Science courses at University of Alberta.

COURSE TITLES FOR FIGS. 1, 2, 3.

Courses in first (2XX), second (3XX), and third (4XX) year only are included.

General
214: Introduction to Computing*
215: Computers and Programming*
314: Introduction to Discrete Structures*

Stream A: Numerical Mathematics
316: Numerical Calculus*
321: Probability*
322: Probability and Statistics*
418: Numerical Analysis I*
419: Numerical Analysis II*
421: Topics in Sampling and Simulation.

Stream B: Systems and Languages
315: Computer Organisation and Data Structures
413: Telecommunications and Computers
414: Programming Languages*
415: Compiler Construction*
416: Systems Programming*
417: Basic File Management
463: Structure of Digital Machines

Stream C: Formal and Adaptive Systems
462: Sequential Machines
464: Introduction to Adaptive Systems
465: Threshold Logic and Learning Machines

Introductory Service courses
300: Problem Solving and Programming (Engineering)
305: Computer Methods for Scientists (Physics)
310: Elements of Programming (Arts and Education)
459: Introduction to Scientific Programming
(Agriculture, Bilogy, Sociology)

Later Service courses
400: Advanced Programming I
405: Advanced Programming II
440: Linear Programming and Matrix Algebra
458: Numerical Analysis for Engineers

*Course title from Curriculum 68. Our own title and course content may be slightly different.

3) To specify programs in Computing Science which may be appropriate for students in other disciplines.

Since the pertinent courses are under our control, we shall discuss the third of these objectives in the following section.

5. SERVICE PROGRAMS

In conjunction with attempts to define coherent programs in other disciplines for our own students, we would like to simplify access to our own program for students who are not primarily computing scientists. That is, computing science might, in a joint program, be a minor rather than a major subject of concentration. One may feel uneasy, as does Zadeh [5], at this stage of development of the discipline, about graduating a very large number of computing science specialists. However, there is no doubt of the value of a basic education in, and some understanding of, computing science for undergraduates in almost any discipline.

Somewhat surprisingly, we find it easier to specify programs in other departments for our own students than to specify programs in our own department for students from outside the department. One difficulty is that our service courses, (of which we have several for different groups of students) were designed without thought as to what courses might follow for those students who became particularly interested in computing science. Sterling and Pollack [6] mentioned the possibility of students being "trapped" in terminal service courses and, for many, this is exactly what has happened.

Another problem is that our own program includes, in the first and second years, material considered basic by computing scientists - assembler language and discrete mathematics, for example. Though the importance of this material is obvious to the faculty, its relevance is questioned particularly by students in other disciplines who wish to take only four or five courses in computing science. Schwenkel [7], in a discussion of the Notre Dame curriculum, singled out the above areas as those for which it was most difficult to motivate computing science students. It is evident then that the popularity and/or utility of sets of service courses will depend upon the treatment of "basic" computing science material.

At the same time, we are not prepared to postpone this material within the computing science core, yet for practical reasons we must use in service programs some courses developed originally for computing science students. It now appears that a feasible solution is for us to classify students according to

a) the strength of their background in mathematics, and
b) the expected nature of their interest in computing science.

Our present introductory service courses are designed for students in:

a) A Physical Science
b) Engineering
c) Agricultural, Biological, or Social Science
d) Arts and Education
e) Business Administration*

(*This course is given by the Faculty of Business, but their faculty would like students taking it to have further courses available).

Enrolments in these courses range from 80 to 200.

Students taking courses (a) and (b) have strong backgrounds in Mathematics and, if interested in continuing, usually (though not always) want more courses in computational mathematics. We do have a course in Numerical Analysis for engineers which can be taken after an introductory programming course, and which will then allow access to all numerical mathematics courses from third year onwards. This service program is illustrated in Figure 2. The only requirement is that the instructors teaching Numerical Analysis for engineers be aware that their course may be either terminal or preparation for subsequent courses.

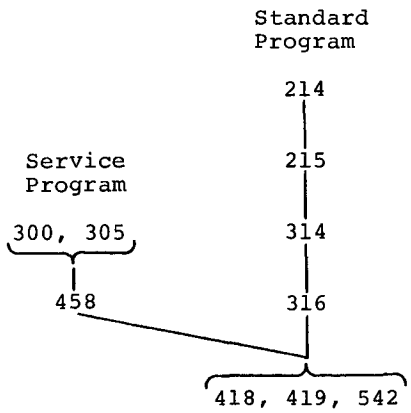

Figure 2. Service Course Program for Numerical Mathematics.

For other students, the solution is not so easily found. As a typical case, let us consider a student who has taken the introductory course in Business Administration and who wishes to do further work in systems and languages. At the moment, he would have to take, in order, the courses in Computers and Programming, Discrete Structures, and Computer Organization and Data Structures before gaining

access to the program of third-year courses. Among the latter are the courses which most attract him - for example, Systems Programming, Telecommunications and Computers, File Management, and Programming Languages. In the case of this prototypical student, he would have taken his introductory course in second year. Since the other three preparatory courses are given in sequence, he may not be qualified to take our third-year courses until he has completed his fourth year.

A solution appears to be in the design of one or more "bridging" courses for students with a given background which will give direct access to a defined subset of later courses. The first attempt at this has resulted in the definition of two courses containing the most relevant material from the three preparatory courses mentioned earlier. The bridging courses now give access to the four later courses as shown in Figure 3. Either of them may also be terminal.

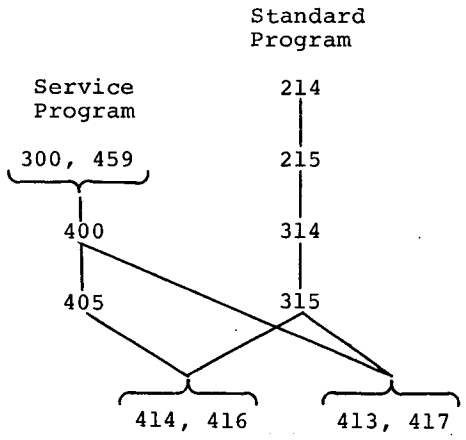

Figure 3. Service Course Program for Systems and Languages.

This arrangement does present complications, in that the instructor of a later course may feel obliged to "dilute" his course to make it more acceptable to students without the complete set of preparatory courses. To minimize this problem, a topic outline for each course is carefully prepared and the instructor should adhere as closely as possible to these outlines. As this attempt at a solution to the service program problem is just being implemented, the final analysis of its success must await experience.

CONCLUSIONS

The combination of a diverse faculty and a large undergraduate population in computing science has created problems of identity and relevance at the University of Alberta. Our general solution to these problems has been flexibility. With the evolving nature of the discipline and the diversity of faculty interests, we have begun to segment the undergraduate curriculum into subprograms while maintaining only a core of relevant material in common. This appears to be a logical evolutionary process having precedents in many other disciplines of science and engineering.

As the study and application of computer systems becomes even more pervasive, we have concluded that the discipline must look outward rather than inward if it is to remain a viable academic discipline. All students who are interested in some facet of computing science must be given every opportunity to pursue that interest. Programs designed for computing science students should have sufficient flexibility to provide a computing science core of lasting value plus basic information in other related disciplines. However, this does not mean that computing science programs can be unilateral and monolithic in character. Students in other disciplines must be provided access to relevant computing science material even when their interest arises late in their program of study. A means of providing this opportunity is accelerated service courses leading into courses originally designed for computing science students only.

We are well aware that these solutions may create the problems of fragmentation and dilution of the curriculum which could be even more knotty than the original problems. However, we firmly believe that an awareness of these conditions coupled with a continuous examination of our computing science programs will provide us with the means to continue a vigorous growth in an evolving discipline.

ACKNOWLEDGEMENTS

The Department of Computing Science has seventeen members, and a majority of these have at one time or another made contributions to curriculum development, in several cases very substantial contributions for which the authors of this paper must give due credit.

REFERENCES

1. Wegner, P. "Some Thoughts on Graduate Education in Computer Science". SIGSCE Bulletin, Volume 2, No. 4, September - October, 1970.

2. Canadian Information Processing Society. CIPS 1971 Curriculum Survey Universities.

3. Amarel, S. "Computer Science: A Conceptual Framework for Curriculum Planning." Comm. ACM, Volume 14, No. 6, June, 1971.

4. ACM Curriculum Committee. "Curriculum 68 - Recommendations for Academic Programs in Computer Science." Comm. ACM, Volume 11, No. 3, March, 1968.

5. Zadeh, L.A. "The Dilemma of Computer Sciences", University Education in Computing Science, edited by A. Finerman, Academic Press, New York, 1968.

6. Sterling, T., and S. Pollack, "Experience with a 'Universal' Introductory Course in Computer Science", SIGCSE Bulletin, Volume 2, No. 3, November 16, 1970.

7. Schwenkel, F. "Remarks on the Notre Dame Computer Science Curriculum." SIGSCE Bulletin, Volume 3, No. 1, March, 1971.