LIVING WITH A NEW MATHEMATICAL SPECIES

Lynn Arthur Steen
St. Olaf College, Northfield, Minn., 55077, U.S.A.

Computers are both the creature and the creator of mathematics. They are, in the apt phrase of Seymour Papert, "mathematics-speaking beings". More recently J. David Bolter in his stimulating book Turing's Man [4] calls computers "embodied mathematics". Computers shape and enhance the power of mathematics, while mathematics shapes and enhances the power of computers. Each forces the other to grow and change, creating, in Thomas Kuhn's language, a new mathematical paradigm.

Until recently, mathematics was a strictly human endeavor. But suddenly, in a brief instant on the time scale of mathematics, a new species has entered the mathematical ecosystem. Computers speak mathematics, but in a dialect that is difficult for some humans to understand. Their number systems are finite rather than infinite; their addition is not commutative; and they don't really understand "zero", not to speak of "infinity". Nonetheless, they do embody mathematics.

The core of mathematics is changing under the ecological onslaught of mathematics-speaking computers. New specialties in computational complexity, theory of algorithms, graph theory, and formal logic attest to the impact that computing is having on mathematical research. As Arthur Jaffe has argued so well (in [12]), the computer revolution is a mathematical revolution.

New Mathematics for a New Age

Computers are discrete, finite machines. Unlike a Turing machine with an infinite tape, real machines have limits of both time and space. Theirs is not an idealistic Platonic mathematics, but a mathematics of limited resources. The goal is not just to get a result, but to get the best result for the least effort. Optimization, efficiency, speed, productivity--these are essential objectives of modern computer mathematics.

Computers are also logic machines. They embody the fundamental engine of mathematics--rigorous propositional calculus. The first celebrated computer proof was that of the four-color theorem: the computer served there as a sophisticated accountant, checking out thousands of cases of reductions. Despite philosophical alarms that computer-based proofs

change mathematics from an *a priori* to a contingent, fallible subject (see, e.g., [27]), careful analysis reveals that nothing much has really changed. The human practice of mathematics has always been fallible; now it has a partner in fallibility.

Recent work on the mysterious Feigenbaum constant reveals just how far this evolution has progressed in just eight years: computer-assisted investigations of families of periodic maps suggested the presence of a mysterious universal limit, apparently independent of the particular family of maps. Subsequent theoretical investigations led to proofs that are true hybrids of classical analysis and computer programming [8], showing that computer-assisted proofs are possible not just in graph theory, but also in functional analysis.

Computers are also computing machines. By absorbing, transforming, and summarizing massive quantities of data, computers can simulate reality. No longer need the scientist build an elaborate wind tunnel or a scale model refinery in order to test engineering designs. Wherever basic science is well understood, computer models can emulate physical processes by carrying out instead the process implied by mathematical equations. Whereas mathematical models used to be primarily tools used by theoretical scientists to formulate general theories, now they are practical tools of enormous value in the everyday world of engineering and economics.

It has been fifty years since Alan Turing developed his seminal scheme of computability [26], in which he argued that machines could do whatever humans might hope to do. In abstract terms, what he proposed was a universal machine of mathematics (see [11] for details). It took two decades of engineering effort to turn Turing's abstract ideas into productive real machines. During that same period abstract mathematics flourished, led by Bourbaki, symbolized by the "generalized abstract nonsense" of category theory. But with abstraction came power, with rigor came certainty. Once real computers emerged, the complexity of programs quickly overwhelmed the informal techniques of backyard programmers. Formal methods became *de rigueur*; even the once-maligned category theory is now enlisted to represent finite automata and recursive functions (see, e.g., [2]). Once again, as happened before with physics, mathematics became more efficacious by becoming more abstract.

The Core of the Curriculum

Twenty years ago in the United States the Committee on the Undergraduate Program in Mathematics (CUPM) issued a series of reports that led to a gradual standardization of curricula among undergraduate mathematics departments [5]. Shortly thereafter, in 1971, Garrett Birkhoff and J. Barkley Rosser presented papers at a meeting of the Mathematical Association of America concerning predictions for undergraduate mathematics in 1984. Birkhoff urged increased emphasis on modelling, numerical algebra, scientific computing, and discrete mathematics. He also advocated increased use of computer methods in

pure mathematics: "Far from muddying the limpid waters of clear mathematical thinking, [computers] make them more transparent by filtering out most of the messy drudgery which would otherwise accompany the working out of specific illustrations." [3, p. 651] Rosser emphasized many of the same points, and warned of impending disaster to undergraduate mathematics if their advice went unheeded: "Unless we revise [mathematics courses] so as to embody much use of computers, most of the clientele for these courses will instead be taking computer courses in 1984." [21, p. 639]

In the decade since these words were written, U.S. undergraduate and graduate degrees in mathematics have declined by 50%. The clientele for traditional mathematics has indeed migrated to computer science, and the former CUPM consensus is all but shattered. Five years ago CUPM issued a new report, this one on the Undergraduate Program in Mathematical Sciences [6]. Beyond calculus and linear algebra, they could agree on no specific content for the core of a mathematics major: "There is no longer a common body of pure mathematical information that every [mathematics major] should know."

The symbol of reformation has become discrete mathematics. Several years ago Anthony Ralston argued forcefully the need for change before both the mathematics community [17] and the computer science community [18]. Discrete mathematics, in Ralston's view, is the central link between the fields. The advocacy of discrete mathematics rapidly became quite vigorous (see, e.g., [19] and [24]), and the Sloan Foundation funded experimental curricula at six institutions to encourage development of discrete-based alternatives to standard freshman calculus.

The need for consensus on the contents of undergraduate mathematics is perhaps the most important issue facing American college and university mathematics departments. On the one hand departments that have a strong traditional major often fail to provide their students with the robust background required to survive the evolutionary turmoil in the mathematical sciences. Like the Giant Panda, these departments depend for survival on a dwindling supply of bamboo--strong students interested in pure mathematics. On the other hand, departments offering flabby composite majors run a different risk: by avoiding advanced, abstract requirements, they often misrepresent the true source of mathematical knowledge and power. Like zoo-bred animals unable to forage in the wild, students who have never been required to master a deep theorem are ill-equipped to master the significant theoretical complications of real-world computing and mathematics.

Computer Literacy

Mathematical scientists at American institutions of higher education are responsible not only for the technical training of future scientists and engineers, but also for the technological literacy of laymen--of future lawyers, politicians, doctors, educators, and clergy. Public demand that college graduates be prepared to live and work in a

computer age has caused many institutions to introduce requirements in quantitative or computer literacy.

In 1981 the Alfred P. Sloan foundation initiated curricular exploration of "the new liberal arts", the role of applied mathematical and computer sciences in the education of students outside technical fields. "The ability to cast one's thoughts in a form that makes possible mathematical manipulation and to perform that manipulation ... [has] become essential in higher education, and above all in liberal education." [14, p. 6] Others echoed this call for reform of liberal education. David Saxon, President of the University of California wrote in a Science editorial that liberal education "will continue to be a failed idea as along as our students are shut out from, or only superficially acquainted with, knowledge of the kinds of questions science can answer and those it cannot." [22]

Too often these days the general public views computer literacy as a modern substitute for mathematical knowledge. Unfortunately, this often leads students to superficial courses that emphasize vocabulary and experiences over concepts and principles. The advocates of computer literacy conjure images of an electronic society dominated by the information industries. Their slogan of "literacy" echoes traditional educational values, conferring the aura but not the logic of legitimacy.

Typical courses in computer literacy are filled with ephemeral details whose intellectual life will barely survive the students' school years. These courses contain neither a Shakespeare nor a Newton, neither a Faulkner nor a Darwin; they convey no fundamental principles nor enduring truths. Computer literacy is more like driver education than like calculus. It teaches students the prevailing rules of the road concerning computers, but does not leave them well prepared for a lifetime of work in the information age.

Algorithms and data structures are to computer science what functions and matrices are to mathematics. As much of the traditional mathematics curriculum is devoted to elementary functions and matrices, so beginning courses in computing--by whatever name--should stress standard algorithms and typical data structures. As early as students study linear equations they could also learn about stacks and queues; when they move on to conic sections and quadratic equations, they could in a parallel course investigate linked lists and binary trees.

Computer languages can (and should) be studied for the concepts they represent--procedures in Pascal, recursion and lists for Lisp--rather than for the syntactic details of semicolons and line numbers. They should not be undersold as mere technical devices for encoding problems for a dumb machine, nor oversold as exemplars of a new form of literacy. Computer languages are not modern equivalents of Latin or French; they do not deal in nuance and emotion, nor are they capable of persuasion, conviction, or humor. Although computer languages do represent a new and powerful way to think about problems, they are not a new form of literacy.

Computer Science

In the United States, computer science programs cover a broad and varied spectrum, from business-oriented data processing curricula, through management information science, to theoretical computer science. All of these intersect with the mathematics curriculum, each in different ways.

Recently Mary Shaw of Carnegie Mellon University put together a composite report on the undergraduate computer science curriculum. This report is quite forceful about the contribution mathematics makes to the study of computer science: "The most important contribution a mathematics curriculum can make to computer science is the one least likely to be encapsulated as an individual course: a deep appreciation of the modes of thought that characterize mathematics." [23. p. 55]

The converse is equally true: one of the more important contributions that computer science can make to the study of mathematics is to develop in students an appreciation for the power of abstract methods when applied to concrete situations. Students of traditional mathematics used to study a subject called "Real and Abstract Analysis"; students of computer science now can take a course titled "Real and Abstract Machines". In the former "new math", as well as in modern algebra, students learned about relations, abstract versions of functions; today business students study "relational data structures" in their computer courses, and advertisers tout "fully relational" as the latest innovation in business software.

An interesting and pedagogically attractive example of the power of abstraction made concrete can be seen in the popular electronic spreadsheets that are marketed under such trade names as Lotus and VisiCalc. Originally designed for accounting, they can as well emulate cellular automata or the Ising model for ferromagnetic materials [10]. They can also be "programmed" to carry out most standard mathematical algorithms--the Euclidean algorithm, the simplex method, Euler's method for solving differential equations [1]. An electronic spreadsheet--the archetype of applied computing--is a structured form for recursive procedures--the fundamental tool of algorithmic mathematics. It is a realization of abstract mathematics, and carries with it much of the power and versatility of mathematics.

Computers in the Classroom

Just as the introduction of calculators upset the comfortable pattern of primary school arithmetic, so the spread of computers will upset the traditions of secondary and tertiary mathematics. This year long division is passe; next year integration will be under attack.

The impact of computing on secondary school mathematics has been the subject of many recent discussions in the United States. Jim Fey, coordinator of two of the most recent assessments ([7], [9]), described these efforts as "an unequivocal dissent from the spirit and substance

of efforts to improve school mathematics that seek broad agreement on conservative curricula." [9, p. viii] Teachers in tune with the computer age seek change in both curriculum and pedagogy. But the inertia of the system remains high. For example, the recent International Assessment of Mathematics documented that in the United States calculators are never permitted in one-third of the 8th grade classes, and rarely used in all but 5% of the rest [25, p. 18].

Lap size computers are now common--they cost about as much as ten textbooks, but take up only the space of one. Herb Wilf argues (in [28]) that it is only a matter of time before students will carry with them a device to perform all the algorithms of undergraduate mathematics. Richard Rand, in a survey [20] of applied research based on symbolic algebra agrees: "It will not be long before computer algebra is as common to engineering students as the now obsolete slide rule once was."

Widespread use of computers that do school mathematics will challenge standard educational practice. For the most part, computers reinforce the student's desire for correct answers. In the past, their school uses have primarily extended the older "teaching machines": programmed drill with pre-determined branches for all possible responses. But the recent linking of symbolic algebra programs with so-called "expert systems" into sophisticated "intelligent tutors" has produced a rich new territory for imaginative computer-assisted pedagogy that advocates claim can rescue mathematics teaching from the morass of rules and template-driven tests.

It is commonplace now to debate the wisdom of teaching skills (such as differentiation) that computers can do as well or better than humans. Is it really worth spending one month of every year teaching half of a country's 18 year old students how to imitate a computer? What is not yet so common is to examine critically the effect of applying to mathematics pedagogy computer systems that are themselves only capable of following rules or matching templates. Is it wise to devise sophisticated computer systems to teach efficiently precisely those skills that computers can do better than humans, particularly those skills that make the computer tutor possible? In other words, since computers can now do the calculations of algebra and calculus, should we use this power to reduce the curricular emphasis on calculations or to make the teaching of these calculations more efficient? This is a new question, with a very old answer.

Let Us Teach Guessing

35 years ago George Polya wrote a brief paper with the memorable title "Let Us Teach Guessing" [16]. It is not differentiation that our students need to learn, but the art of guessing. A month spent learning the rules of differentiation reinforces a student's ability to learn (and live by) the rules. In contrast, time spent making conjectures about derivatives will teach a student something about the art of mathematics and the science of order.

With the aid of the mathematics-speaking computer, students can for the first time learn college mathematics by discovery. This is an opportunity for pedagogy that mathematics educators cannot afford to pass up. Mathematics is, after all, the science of order and pattern, not just a mechanism for grinding out formulas. Students discovering mathematics gain insight into the discovery of pattern, and slowly build confidence in their own ability to understand mathematics. Formerly, only students of sufficient genius to forge ahead on their own could have the experience of discovery. Now with computers as an aid, the majority of students can experience for themselves the joy of discovery.

Metaphors for Mathematics

Two metaphors from science are useful for understanding the relation between computer science, mathematics, and education. Cosmologists long debated two theories for the origin of the universe--the Big Bang theory, and the theory of Continuous Creation. Current evidence tilts the cosmology debate in favor of the Big Bang. Unfortunately, this is all too often the public image of mathematics as well, even though in mathematics the evidence favors Continuous Creation.

The impact of computer science on mathematics and of mathematics on computer science is the most powerful evidence available to beginning students that mathematics is not just the product of an original Euclidean big bang, but is continually created in response to challenges both internal and external. Students today, even beginning students, can learn things that were simply not known 20 years ago. We must not only teach new mathematics and new computer science, but we must teach as well the fact that this mathematics and computer science is new. That s a very important lesson for laymen to learn.

The other apt metaphor for mathematics comes from the history of the theory of evolution. Prior to Darwin, the educated public believed that forms of life were static, just as the educated public of today assumes that the forms of mathematics are static, laid down by Euclid, Newton and Einstein. Students learning mathematics from contemporary textbooks are like the pupils of Linnaeus, the great eighteenth century Swedish botanist: they see a static, pre-Darwinian discipline that is neither growing nor evolving. Learning mathematics for most students is an exercise in classification and memorization, in labelling notations, definitions, theorems, and techniques that are laid out in textbooks as so much flora in a wonderous if somewhat abstract Platonic universe.

Students rarely realize that mathematics continually evolves in response to both internal and external pressures. Notations change; conjectures emerge; theorems are proved; counterexamples are discovered. Indeed, the passion for intellectual order combined with the pressure of new problems--especially those posed by the computer--force researchers to continually create new mathematics and archive old theories.

The practice of computing and the theory of computer science combine to change mathematics in ways that are highly visible and attractive to students. This continual change reveals to students and laymen the living character of mathematics, restoring to the educated public some of what the experts have always known--that mathematics is a living, evolving component of human culture.

REFERENCES

1. Arganbright, Dean E. Mathematical Applications of Electronic Spreadsheets. McGraw-Hill, 1985.
2. Beckman, Frank S. Mathematical Foundations of Programming. The Systems Programming Series, Addison Wesley, 1984.
3. Birkhoff, Garrett. "The Impact of Computers on Undergraduate Mathematics Education in 1984." American Mathematical Monthly 79 (1972) 648-657.
4. Bolter, J. David. Turing's Man: Western Culture in the Computer Age. University of North Carolina Press, Chapel Hill, 1984.
5. Committee on the Undergraduate Program in Mathematics. A General Curriculum in Mathematics for Colleges. Mathematical Association of America, 1965.
6. Committee on the Undergraduate Program in Mathematics. Recommendations for a General Mathematical Sciences Program. Mathematical Association of America, 1980.
7. Corbitt, Mary Kay, and Fey, James T. (Eds.). "The Impact of Computing Technology on School Mathematics: Report of an NCTM Conference." National Council of Teachers of Mathematics, 1985.
8. Eckmann, J.-P., Koch, H., and Wittwer, P. "A computer-assisted proof of universality for area-preserving maps." Memoirs of the American Mathematical Society, Vol 47, No. 289 (Jan. 1984).
9. Fey, James T., et al. (Eds.). Computing and Mathematics: The Impact on Secondary School Curricula. National Council of Teachers of Mathematics, 1984.
10. Hayes, Brian. "Computer Recreations." Scientific American (October 1983) 22-36.
11. Hodges, Andrew. Alan Turing: The Enigma. Simon and Schuster, 1983.
12. Jaffe, Arthur. "Ordering the Universe: The Role of Mathematics." In Renewing U. S. Mathematics, National Academy Press, Washington, D. C. 1984.
13. Kemeny, John G. "Finite Mathematics--Then and Now." In Ralston, Anthony and Young, Gail S. The Future of College Mathematics. Springer-Verlag, 1983, pp. 201-208.
14. Koerner, James D., ed. The New Liberal Arts: An Exchange of Views. Alfred P. Sloan Foundation, 1981.
15. Lewis, Harry R. and Papadimitriou. Elements of the Theory of Computation. Prentice-Hall. 1981.
16. Polya, George. "Let Us Teach Guessing." Etudes de Philosophie des

Sciences. Neuchatel: Griffon, 1950, pp. 147-154; reprinted in George Polya: Collected Papers. Vol. IV, MIT Press, 1984, pp. 504-121.

17. Ralston, Anthony. "Computer Science, Mathematics, and the Undergraduate Curricula in Both." American Mathematical Monthly 88 (1981) 472-485.
18. Ralston, Anthony and Shaw, Mary. "Curriculum '78: Is Computer Science Really that Unmathematical?" Communications of the ACM 23 (Feb. 1980) 67-70.
19. Ralston, Anthony and Young, Gail S. The Future of College Mathematics. Springer Verlag, 1983.
20. Rand, Richard H. Computer Algebra in Applied Mathematics: An Introduction to MACSYMA. Research Notes in Mathematics No. 94, Pitman Publ., 1984.
21. Rosser, J. Barkley. "Mathematics Courses in 1984." American Mathematical Monthly 79 (1972) 635-648.
22. Saxon, David S. "Liberal Education in a Technological Age." Science 218 (26 Nov 1982) 845.
23. Shaw, Mary (Ed.) The Carnegie-Mellon Curriculum for Undergraduate Computer Science. Springer Verlag, 1984.
24. Steen, Lynn Arthur. 1 + 1 = 0: New Math for a New Age. Science 225 (7 Sept. 1984) 981.
25. Travers, Kenneth, et. al. Second Study of Mathematics: United States Summary Report. University of Illinois, September 1984.
26. Turing, Alan M. "On Computable Numbers, with an Application to the Entscheidungsproblem." Proc. London Math. Soc. 2nd Ser., 42 (1936) 230-265.
27. Tymoczko, Thomas. "The Four Color Problem and its Philosophical Significance." Journal of Philosophy 76:2 (1979) 57-85.
28. Wilf, Herbert. "The Disk with the College Education." American Mathematical Monthly 89 (1982) 4-8.