

The Applied Mathematics and Computer Science Schism

Janusz Kowalik



Isolating applied mathematics from computer science harms the profession and those who depend on it.

Sociologists tell us we evolved from a culture of generalists to one of specialists during the latter half of the 20th century. We see this trend in medicine, finance, manufacturing, and our personal shopping habits, in which customized products appeal to individual needs.

This development stems in part from the staggering capability of information technology to process and manipulate data. For the most part, we view this capacity as positive and admirable. But we forget that specialization often leads to fragmentation and a subsequent failure to thrive.

Programming and logic go hand in hand. Computational science can be regarded as a combination of mathematics, computing, and various application fields. But there is a worrisome trend toward the isolation of mathematics from computing, both at universities and in the computing profession.

Given the widespread availability of software tools such as Mathematica,

AutoCAD, Excel, and others, it can be argued that few computing professionals need mathematical knowledge. But using black boxes without understanding their contents prevents intelligent interpretation of the computed results. Worse, it could be dangerous. Who would want to fly in an airplane whose designers did not fully comprehend the mathematical tools used?

REFOCUSING ON MATHEMATICS

There are compelling reasons for refocusing on the benefits of mathematical training within the computing field. First, mathematics teaches us the valuable skill of reasoning rigorously and proving propositions. Computing professionals with mathematical skills thus think more rationally and can argue more convincingly when espousing their products' and services' benefits.

For example, vendors commonly claim that new computer hardware is *scalable*, thus it is possible to solve increasingly larger problems by

adding more memory or processors. Such statements typically lack valid proofs or empirical evidence.

Strictly speaking, scalability addresses computer capacity vis-à-vis the workload the system is processing. A given computer architecture running a given workload scales if the computation retains efficiency with increasing workload and increasing computer configuration. This requires a good definition of computational efficiency.

Whereas the computing profession well understands efficiency, computational scalability remains an elusive concept, even though it can be understood with even a modicum of mathematically oriented reasoning. In today's prevailing computing culture, scalability is an empty slogan that misleads naïve customers who lack numerical skills and therefore cannot ask the pertinent questions. The blind thus lead the blind.

Second, because of their dwindling availability, it has become difficult to assess the actual need for mathematically oriented computing professionals. Further, managers—who influence the demand—do not generally appreciate the potential usefulness of mathematical modeling and analysis.

This creates the classic vicious circle of supply and demand. The primary victims are high-tech companies and computer science students—most of whom, by virtue of their training, know little of mathematics and are often hostile toward it.

This attitude is notably absent in less developed countries whose educational systems embrace a more traditional mathematical culture. In countries such as Russia, Poland, and Italy, master's theses in informatics frequently incorporate solid mathematical content.

Despite the undeniable leadership of the US in computing technology, we are becoming a country in which most PCs are used for playing games, watching movies, and pursuing other hedonistic gratification. Their owners regard them as fun gadgets that can happily lure them away from reading

Continued on page 102

Continued from page 104

or other activities that foster intellectual growth.

If this negative trend continues, the US will be unable to sustain its technological leadership. Yes, contemporary America has great inventors, scientists, and entrepreneurs, but the average Joe is increasingly illiterate and numerically challenged. Joe likes to have fun, and he leaves the hard thinking to the young immigrants who win spelling bees, play chess, and use the Internet to find proofs of the 2,000-year-old Pythagorean theorem. Joe has watched his television tube for thousands of hours throughout his lifetime, but none of his heroes knows who Pythagoras was. If Joe happens to be a carpenter, he doesn't realize how this theorem can help him in his work.

Despite these current trends in society generally and in computing technology particularly, we must reject the argument that we have no need for mathematically trained computing professionals who can use computers to model and solve real-life problems. Opportunity areas include medicine, finance, economics, space, network security, and global demographics.

CHALLENGES

As scientific knowledge inexorably increases, we face a continuous stream of new problems to solve. Some of these problems challenge us on a grand scale, while others require fewer resources. Grand challenges include space exploration, hurricane path prediction, global climate modeling, drug design, supernova analysis, and protein folding. Others include understanding biological evolution, galaxy formation, the DNA programming language, cross-field computation as required for aeroelasticity design, the nature of genetic diseases, and the origin and evolution of the universe.

Additionally, we face less difficult challenges that nevertheless demand efficient algorithms and fast computers: computational fluid dynamics, structural crash analysis, complex aircraft analysis, factory scheduling, and financial modeling.

Grand challenges pose fundamental problems in science, engineering, and business that cannot be solved without advanced mathematical modeling and large computational resources. The mathematical tools required for solving these problems will vary from case to case. Great advances—such as sparse linear algebra—have been made with mathematical approaches and software for solving very large problems. Computational power on a correspondingly grand scale can be achieved by exploiting supercomputers such as NASA's Columbia system or using grid computing technology.

Solving intricately complex grand challenges will require new, efficient algorithms and methods.

Despite the availability of these mathematical and computational tools, it would be wrong to assume that we will always be able to solve arbitrarily large problems simply by pulling together enough computer resources. Solving intricately complex grand challenges will require new, efficient algorithms and methods. This in turn points to the need for highly skilled computing professionals with strong mathematics skills: Without them, grand-challenge problems will remain unsolved, and lesser problems will be approached from a limited perspective bereft of mathematical insight.

Even though only some computing professionals will confront these types of problems, computer science training must include the mathematics foundation that will let them grasp the intricacy of each problem and propose solutions. As a rule, the greater our mathematical understanding of a complex problem, the more we will benefit from its solution. Thus it follows that our educational system must

create opportunities for the next generation of challenge solvers by including solid mathematics training throughout the curriculum, especially in secondary schools and at the college and university level. Only through education can we reverse the current trend in computer science and technology.

SEVERED DISCIPLINES


Although the roots of the problem lie in basic schooling, academia has a hefty share in creating this deplorable situation.

At most US universities, separate departments teach applied mathematics and computer science. Students interested in these fields must choose one curriculum or the other, but cannot easily integrate them in pursuit of a degree, especially an advanced degree. Not only does this prevent students from acquiring the complementary mathematical skills needed to enhance their expertise in computer science, it also costs employers dearly. How many computer science graduates have never heard of fast Fourier transform or advanced linear algebra algorithms, two very powerful tools used for numerous industrial and business computer applications?

On the other hand, a student of applied mathematics who enters the work force would almost certainly not know about parallel computing, the message passing interface, or grid computing technology. Companies that need quantitative problem modelers and problem solvers must hire both types of underdeveloped graduates and hope they will jointly have enough knowledge to solve the problems they encounter.

While not an unreasonable proposition, this is definitely an expensive one. It diminishes the productivity of younger employees and requires extra effort on their part to acquire the necessary complementary skills.

Some large companies, such as Boeing, have successfully blended and optimized their investment in mathematics and computer science graduates by immersing them in a large



organization in which they close the gaps in their knowledge and become fully productive. Smaller companies can't afford this approach and would much prefer to employ fewer individuals with sufficient skills to build mathematical models, identify efficient solution algorithms, and implement them on proper computers.

Fixing it

Given that the roots of the problem lie with our current educational system, the solution ought to be found in the same place.

As a first step, we should create more interdisciplinary programs involving mathematics, computer science, engineering, physics, and related departments. These programs would combine applied mathematics with an emphasis on numerical analysis, programming that includes elements of software engineering, and related fields. Several top universities in the US—including MIT, Princeton, and Brown—have already implemented this approach with great success.

The innovative textbook *Parallel Scientific Computing in C++ and MPI* (G.E. Karniadakis and R.M Kirby II, Cambridge University Press, 2003) provides an excellent example of how to integrate the teaching of mathematics and computer science. Although designed primarily for undergraduate and graduate students of mathematics, computer science, engineering, and physics, this book can also benefit mathematically oriented students in the biological sciences, finance, and other fields that rely on mathematical models and reasoning.

Who will benefit from this kind of education? First, students trained with this interdisciplinary approach will be more employable than their traditionally trained counterparts. They will have the inside track on better-paying, more challenging jobs. Second, their employers will obtain efficient workers capable of handling mathematical problems, from mathematical model definition to quantitative analysis and numerical results.

Sound advice

Mike O'Neal presented excellent arguments for restructuring computing programs to meet employment challenges in his 2004 essay, "Restructuring Computing Programs to Meet Employment Challenges" (*Computer*, Nov. 2004, pp. 29-34). Simone Santini presented a complementary viewpoint, "Determining Computing Science's Role" (*Computer*, Dec. 2004, pp. 128, 126-127), and observed that universities should stimulate students' intellectual curiosity and provide them the necessary cultural awareness—both general and specific to a discipline.

Combining mathematical and computer science training would help accomplish both objectives by making students more useful for employment and giving them a universal tool for understanding and solving new problems.

Another educator, Paul Rosenbloom, suggested that computer studies should be based on interdisciplinary approaches with a system-oriented perspective in "A New Framework for Computer Science and Engineering" (*Computer*, Nov. 2004, pp. 23-28). Using Paul's notation, my thesis—Computing, Mathematics, and Application Discipline—offers the best approach to education, research, and the application of computer science.

Allen Tucker and colleagues offered a similar recommendation in "Why Math?" (*Comm. ACM*, Sept. 2003, pp. 40-44). They stated that the mathematical thinking and mathematics taught in a computer science curriculum prepare students for all stages of system development.

Despite the overwhelming benefits of this approach, for many reasons few faculty members implement it. For one thing, the university reward system undervalues interdisciplinary teaching: Publishing a narrowly focused paper earns more regard than teaching an interdisciplinary class. Further, it takes more energy and preparation to develop and teach a course that integrates mathematics with computing and some application

field because many professors are themselves victims of their own traditional education, which has isolated mathematics from computer science.

It may seem that reintroducing mathematics into computer science represents only a small step in restructuring higher education in the technologically most advanced countries. After all, many find mathematics a difficult subject, one that can't be mastered simply by blindly pushing buttons and using canned programs.

But mathematical knowledge offers great rewards. Mathematical processes support our technical arguments and help make the world more understandable. In such a world, we know what computing scalability is. Many intelligent adults admit with pride that they have never liked nor understood mathematics. They do not realize that mathematical ignorance, like illiteracy, undermines their ability to understand the world around them and to function effectively in it. Mathematical knowledge is a powerful tool for most computer-oriented professionals.

Reversing this trend toward compartmentalization and moving toward integrating the two disciplines could prove a lengthy process. The computing profession can take the lead by showing that mathematics rightfully belongs in computing. ■

Janusz Kowalik is an international consultant and freelance educator. Contact him at j.kowalik@comcast.net.

Editor: Neville Holmes, School of Computing, University of Tasmania; neville.holmes@utas.edu.au. Links to further material are at www.comp.utas.edu.au/users/nholmes/prfsn.