"... And Next Year, We're Going to Do 3D Problems"

Francis Sullivan, Editor in Chief

For many years, computational modeling talks ended with a sentence like the one in this essay's title. Serious computational researchers were forced to live the life of a great white shark—silently cruising the shallows, always looking for more computing power, compelled to swim, hunt, and eat continually, or sink to the bottom and slowly die.

When you received a new machine, three things usually happened:

1. You tried the largest problem that could run on the previous machine. ("Wow! This really is faster.")
2. You tried a real problem beyond the previous machine's capabilities. ("Well, it did it, finally.")
3. You tried a wild, far-out problem. ("Okay, so I guess it can't do the complete set of solutions for the 4,096 queens problem.")

This initial period was followed by a couple of productive years tuning the system, refining the physical models, developing new algorithms based on possibilities presented by the new equipment, and, as the endgame, pushing the hardware to its limits. Finally, pressure would be put on management to buy an even newer system, much faster and much cheaper than the previous edition of "the world's fastest machine."

From the scientist's viewpoint, modeling needs drove this cycle. But, in fact, it was probably driven by some not completely understood—but surely complex—interaction among technological innovation, global economics, scientific peer pressure, and algorithmic creativity. The fact that you could always expect a new, faster machine just over the horizon defined, and to some extent still defines, Moore's Law. We all used to say that we had two ways to speed up a computation: think hard, design new algorithms, and optimize the code, or wait two years for a faster machine.

An extreme example of pushing computing hardware all the way to the boundary is Seth Lloyd's now-famous article, "Ultimate Physical Limits to Computation" (*Nature*, vol. 406, no. 1047, 2000). Lloyd shows that no matter what assumptions about availability energy and materials you make, the very laws of physics limit what you can do. According to many experts, the pattern of never-ending development of faster machines is about to, well, end. If this is indeed the case, the effects on scientific computation will be profound.

In the coming months, *Computing in Science & Engineering* will publish a series of articles on the general topic of the bounds on computing. We plan to explore the limits of what we can compute now, what might be computed in the foreseeable future, and which computations will never be possible. These articles will range from how-to pieces, through discussions of computations that currently seem to be hitting the wall, all the way to think pieces on what the limits of computation really are. We hope to reflect on some fundamental questions:

1521-9615/02/$17.00 © 2002 IEEE

- What is physically possible to compute? What are the actual theoretical limits of computing power?
- What is possible to compute? What are the practical limits of computation?
- What is "plan B"? When we really do reach the end of Moore's Law, how will we proceed?
- What are the classes of applications that will always demand as much computing as there is?

Computational science is the core subject of one of the three pillars—experiment, theory, and computation—that support the practice of modern science. To continue obtaining the results needed for research, we must devise new methods and algorithms and understand the relationship between computer architecture and numerical computation much more deeply than we do now.

In a practical sense, advancement of computing power probably depends more on algorithms than on physical limits. Even advances in technology are less important than good algorithms. A famous film director once remarked that high-definition TV is not too interesting if there's nothing worth watching. Computing is a little like that—fast machines should open up new vistas rather than merely allow the same things to be done somewhat faster. However, fundamental advances in algorithms are slow in coming compared to technological advances. In many important cases, we don't even have bounds on the number of operations needed to perform everyday calculations. For example, there is not yet a hard upper bound on the absolute minimum number of operations to multiply two matrices. In another direction, we don't even know if the speedups achieved from quantum computing are not also attainable with better classical algorithms.

The slow rate of discovery of truly fundamental new algorithms is to be expected. The question of modeling physical reality with computation is one of the deepest in the physical sciences. It was, after all, consideration of the issues raised by the famous Einstein-Podolsky-Rosen thought experiment that led Richard Feynman to consider the question of using quantum phenomena to model quantum phenomena. This, in turn, gave birth to quantum computing. I suspect that the next level of algorithm development will build on the enhanced understanding of the relationship between computational complexity and the laws of physics, which, in my view, is the important result of research in quantum computing.

New Editorial Board Member

David M. Beazley is an assistant professor in the Department of Computer Science at the University of Chicago, where he teaches courses on operating systems, compilers, and networks. His research primarily focuses on software architecture and software development tools for computing in science and engineering. He was one of the primary developers of SPaSM, a short-range molecular dynamics program used to study problems in fracture, crack propagation, and dislocation dynamics. He also created SWIG, a specialized C++ compiler that automatically creates extension modules for several programming languages. He has a BA in mathematics from Fort Lewis College, an MS in mathematics from the University of Oregon, and a PhD in computer science from the University of Oregon. He is a member of the IEEE, ACM, and Usenix. Contact him at the Univ. of Chicago, 257B Ryerson Hall, 1100 East 58th St., Chicago, IL 60637-1581; beazley@cs.uchicago.edu; http://systems.cs.uchicago.edu/~beazley.