

Life Scientists Get Their Game Faces On

Borrowing technology from gamers, science speeds up.

By Mike May

April 1, 2008 | A few years ago, Ryan Schneider, chief technology officer at Acceleware in Calgary, Alberta, and his colleagues were struggling to solve complex computing challenges such as biological-image reconstruction. Needing more speed than mere CPUs could provide, Schneider's team tried field programmable gate areas (FPGAs) and even Microsoft's Xbox, finally settling on graphics processing units (GPUs).

It turns out that GPUs accelerate more than games and graphics. GPUs - single-chip processors that provide extensive parallel programming - hit the PC market in 1999, handling graphics tasks, including rendering and transformation. "As compared to conventional CPUs, GPUs consist of high-bandwidth memories and more floating point-hardware units," says Dinesh Manocha, a professor in the department of computer science at the University of North Carolina at Chapel Hill. "For example, a current high-end GPU, such as the NVIDIA G80, has peak performance of 300 GFLOPS [billion floating point operations per second] for rasterization and memory bandwidth of 86 gigabytes per second. That compares to 50 GFLOPS and 6 gigabytes per second for a 3-GHz, multi-core Intel Core 2 CPU." Today's top-selling GPUs come from a trio of California companies: AMD in Sunnyvale, and Intel and NVIDIA, both in Santa Clara.

Beyond graphics rasterization, GPUs can also accelerate computationally intensive problems. "For those applications that map well to parallelization, there can be a tremendous speedup with little or no cost for hardware," says Jack Collins, manager of scientific computing and program development at the Advanced Biomedical Computing Center, which is run by SAIC for the National Cancer Institute in Frederick, Maryland. "This turns out to be highly beneficial since most every computer has a graphics card with GPUs that can be utilized." In short, GPU computing puts high-performance on almost anyone's desk.

Kudos for New Codes

Getting more than games or graphics rasterization out of GPUs, though, was initially a challenge. Programmers had to use application-programming interfaces (APIs), such as OpenGL or DirectX, to get general computing from a GPU. Developers also needed to understand the details and organization of GPU hardware. Moreover, an algorithm only runs fast on a GPU if a programmer modifies the algorithm and data structures to take advantage of a GPU's parallelism.

As Manocha says, "In order to utilize the computational power and memory bandwidth available in the GPUs, many novel algorithms have been designed that utilize the rasterization capabilities for non-graphics applications." Even better, scientists can now interact with a GPU through higher-level interfaces, including NVIDIA's CUDA (free for download at www.nvidia.com/cuda) and AMD's CTM (also free: ati.amd.com/companyinfo/researcher/resources.html).

Collins, for example, uses CUDA, which NVIDIA describes as a C language development environment. "You do not have to be a hardware expert to use CUDA," says Collins. "With CUDA, GPU programming is now much more accessible to the average programmer than in the past." Still, he says that leveraging the full benefits of GPU computing through CUDA requires restructuring code and probably revising the algorithm for the computation at hand.

Expanding Opportunities

NVIDIA provides various GPU systems. CUDA-enable graphics cards start at just \$35 and run as high as \$1299 for the Tesla C870, which was designed specifically for computation. "With our workstation products," says Andy Keane,

general manager of the GPU computing business unit at NVIDIA, "it's like putting a 32- or 64-node computer on your desk. All you do is connect a cable to the PCI slot." NVIDIA also makes data-center systems.

Keane adds that CUDA goes beyond the GPU alone. "CUDA lets you mix the sequential processing on a CPU with the parallelism of GPUs," he says. "And multicore CPUs make the connection even easier. With more CPUs, you can use more GPUs." Moreover, NVIDIA plans to expand CUDA's capabilities, such as adding a FORTRAN interface.

Acceleware also makes software that works with GPUs. With Acceleware's tools, independent software vendors can bypass the details of GPU programming. Acceleware's hardware-software combinations range from \$10,000-75,000. "We provide an API and a library to software vendors," says Schneider. He mentions that one large pharmaceutical company used Acceleware's libraries to reconstruct images. "With the company's old technology," says Schneider, "a single scan took about an hour and a half to reconstruct, and our product reduced that to 3-4 minutes."

With such scans, pharmaceutical scientists "can perform studies to look at the effects of compounds in a variety of disease models, such as neurological and oncological diseases, in addition to bone and musculoskeletal disorders," adds Alice Ford-Hutchinson, product manager for imaging at Acceleware.

Accelerating Applications

In 2006, Manocha's group used GPUs to take on a well-known database-sorting benchmark called Indy. They used a 3-GHz Pentium IV with a 7800GT NVIDIA graphics card to sort 590 million 100-byte records in under 11 minutes, which was a new record (see: <http://research.microsoft.com/barc/SortBenchmark/>).

GPUs can also be applied to many other bioinformatics tasks, including sequence alignment, pattern matching, and fluorescence microscopy. John Stone, a senior research programmer at the Beckman Institute for Advanced Science and Technology at the University of Illinois at Urbana-Champaign, and his colleagues use GPUs for molecular-dynamic simulations. He also uses CUDA, noting, "It only took about a day of programming to get our first piece of code ported to the GPU." But he adds, "We were using a hand-picked problem that I thought would map well to a GPU, and we are very experienced programmers." If code ports efficiently to a GPU, says Stone, it can speed it up by 30-100 times. "The worst case with GPUs," he says, "is around seven times, and that's not bad."

But whether researchers choose GPU computing, FPGAs, or supercomputers, Stone is adamant that code must move to parallel approaches. "Any algorithm that isn't parallel," says Stone, "is now a dead end."

This article appeared in Bio-IT World Magazine.
Subscriptions are free for qualifying individuals. [Apply Today](#).