



# Biomanycores Repository Seeks to Bridge Bioinformatics Software and GPU Computing

September 18, 2009

**Newsletter:** [BioInform](#)  
[BioInform - September 18, 2009](#)

By [Vivien Marx](#)

**As interest** in the use of graphics processing units for accelerating bioinformatics tasks continues to rise, a team of French researchers is looking to make it easier for users to find and use software that has been optimized for GPU-enabled high-performance computing.

The researchers, members of a bioinformatics group called Sequoia at Lille University, have created a repository for bioinformatics parallel code written in the Open Computing Language, OpenCL, or Nvidia's Compute Unified Device Architecture, CUDA.

The team launched the repository, called [Biomanycores](#), in early 2009. It currently includes GPU-enabled implementations of three bioinformatics packages: SWcuda, a version of the Smith-Waterman algorithm; pknotsRG, an algorithm for predicting pseudoknots in an RNA sequence; and cudaPWM, an algorithm for scanning a position weight matrix against a DNA sequence.

Biomanycore's hosts are hoping that other developers will upload their software to the repository. In order to improve interoperability, all the software on the site so far has interfaces with BioPerl, BioJava, and BioPython.

"We are also looking for help to develop interfaces to Bio\* frameworks," Jean-Stéphane Varré, a Sequoia group member and associate professor at INRIA Lille, told *BioInform*.

GPU computing offers considerable speedups for bioinformatics computing — generally between 3-fold and 50-fold, Varré said.

For example, the SWcuda Smith-Waterman implementation in the repository, developed by researchers at the University of Padova, is about 30 times faster than Smith-Waterman on a CPU, he said. "This implies that you can make use of exact search instead of approximate search for comparing sequences."

The cudaPWM package, meantime, runs 77 times faster on an Nvidia GTX 280 as compared to a CPU.

## Joiners Welcome

The Lille team has been working with GPUs since 2007, and this year began a "partnership" with Nvidia, said Mathieu Giraud, another researcher within the Sequoia group who is affiliated with the French national research institute CNRS. He did not offer further details of the nature of the partnership.

When exploring ways to disseminate algorithms, "we decided the good way was to create a repository" for scientists to upload their own software or to download software from others, Giraud said.

Biomanycores started with two in-house algorithms and the publicly available CUDA implementation of the Smith-Waterman algorithm, which was completed by the team at the University of Padova.

"We plan to continue the integration of new applications in the months to come," Giraud said, adding that likely applications will be in the areas of short-read analysis, RNA secondary structure prediction, and phylogeny.

Second-generation sequencers and other high-throughput technologies are driving "exponential" data growth that is outpacing Moore's law, Varré said. While some biology labs have set up powerful computers or even small grids, installation and maintenance of IT systems is difficult without onsite staff, he said.

"Since parallel computing using many-core processors could be an answer, with Biomanycores we would like to help the community to access the latest methods developed for such processors," Varré said.

Varré added that algorithm developers are not "the primary users" of Biomanycores. As an example of how it might be used in practice, he said that he and his colleagues are working with biologists to develop an analysis pipeline in the field of comparative genomics for non-coding sequence analysis.

"Using GPUs via Biomanycores in this pipeline will help avoid some hurdles," he said. "We think that it is the spirit of Biomanycores: providing very efficient implementations for the most complex or repetitive tasks."

But not everyone agrees that such a resource would be of immediate use for end-user biologists. "Libraries would help to accelerate the implementation process, Bertil Schmidt, a computer scientist at Nanyang Technological University in Singapore, told *BioInform* via e-mail. "However, Biomanycores interfaces are still in early stage of development."

Schmidt developed CudaSW++, to optimize Smith-Waterman sequence database searches on CUDA-enabled GPUs, which was published in *BMC Research Notes*, in May.

Schmidt's group is designing "efficient CUDA solutions" for a variety of bioinformatics applications and has worked on multiple sequence alignment, Smith-Waterman database scanning, motif finding, and molecular dynamics. "This work is important since it provides

around one order-of-magnitude speedup on a single standard GPU," he said.

Schmidt said that he is planning on releasing CUDA-BLASTP and a CUDA-based assembly tool for Illumina short-read data soon. "We probably will publish those on [sourceforge.net](http://sourceforge.net)," he said in response to a question about whether he will deposit it on Biomanycores.

## Be Parallel

Another aim of Biomanycores is to offer recommendations for scientists working with GPU computing. For example, developers should remember to not benchmark algorithms and hardware together — "a common pitfall in parallelism," Giraud said.

For some algorithms, there are "standard metrics," Varré said. For example, Smith-Waterman speed can be measured in MCUPS, millions of cells updated per second. This allows researchers "to compare algorithms across different implementations and hardware."

Good benchmarks should run the same algorithm on different processors, presenting and explaining the bottlenecks in each architecture, Giraud said. "It is also important to describe the nature of the speedups: is it against the same algorithm in a dummy one-core CPU version, or against an optimized CPU version that includes multithreading and [streaming single instruction, multiple data extensions] instructions?"

Although the bar for users in this area is still high, the Biomanycores team believes it is worth the effort.

"A key point is how you access your memory. You usually have fast, but small, caches, and large, but slow, global memories," Giraud said. With some processors, the cache is even explicitly managed: which means researchers must design algorithms that work on small subsets of data.

Varré said that researchers must choose programming models carefully. Current GPU architectures are mostly SIMD, or single instruction, multiple data, processors. "To be fully efficient, you should understand this model and fit your computations to it," he said.

## Going Bio

Biomanycores offers a BioJava API, but scientists need the Java build tool Apache Ant as well as a number of CUDA programs before they can run Biomanycores interfaces.

Users need to install "at least one of the three CUDA programs available in Biomanycores," said Stéphane Janot, another researcher in the Sequoia group. "That is not always easy, and will be one point we want to improve" to allow a better integration between BioJava and the CUDA programs. The team has similar plans for the BioPerl and BioPython APIs.

CUDA is nVidia's proprietary GPU programming model, and the programming language is "C with CUDA extensions," Andrew Humber, nVidia's senior PR manager for CUDA products told *BioInform* via e-mail.

There are many players in the GPU market, notably nVidia, AMD and others. The industry is moving toward adoption of the Open Computing Language, or OpenCL, a standard for "heterogeneous parallel programming" managed by an industry consortium called the Khronos Compute Working Group.

OpenCL is "largely based" on Nvidia's "own C language implementation, although it's slightly higher level," Humber said, adding that the company chairs the working group that helped ratify and release the OpenCL standard.

"Snow Leopard is the first shipping OS to support OpenCL, but we also have drivers for Windows and Linux with developers today," he said. OpenCL runs on Nvidia's CUDA-enabled GPUs.

CUDA, in his view, offers flexibility because it allows the use of "other languages such as Fortran, which still has a great deal of traction in the high-performance computing space" along with Java and Python.

OpenCL stands to offer researchers flexibility in implementing their applications, Varré said. "Indeed, with OpenCL one will be able to develop parallel algorithms and then compile them on several platforms: from multi-core processors, the ones everyone has in their own personal computer today, to many-cores processors such as GPU or, perhaps, the upcoming Intel Larrabee," Varré said.

OpenCL "promises to provide code portability," said Nanyang's Schmidt. "However, it still remains to be seen whether OpenCL implementations are also performance portable; [in that] the same code delivers high performance on a variety of parallel architectures."

Giraud sees it as positive news for the field that the first OpenCL compilers are now available. "Nvidia has one available for developers, Apple has embedded a compiler in OS X Snow Leopard, and AMD/ATI is also releasing a compiler," he said. "In high-performance computing, 2010 should be the year of OpenCL."

#### Genomeweb system

These settings are generally managed by the web site so you rarely need to consider them.

**Issue Order: 2**

