# Biomanycores, a repository of interoperable open-source code for many-cores bioinformatics

Jean-Stéphane Varré Stéphane Janot Mathieu Giraud

LIFL - UMR CNRS 8022, INRIA, Université Lille 1

jean-stephane.varre@lifl.fr, stephane.janot@lifl.fr, mathieu.giraud@lifl.fr

**URL:** `http://www.biomanycores.org`
**Licences:** Various open-source licences

Graphic processing units (GPUs) are a first step towards massively many-cores architectures, and recent trends blur the line between such GPUs and multi-core processors. Those architectures enable efficient parallel processing at a very low cost. The current GPUs offer coarse-grained level parallelism (*work-groups* or *blocks* of independent computations) as well as SIMD-like parallelism (*work-items* or *threads*).

GPUs have been used in bioinformatics since 2005, at the beginning tweaking graphics primitives [3, 8]. The CUDA libraries, first released in 2007 [2], have deeply simplified the development on GPUs. In the two past years a growing number of applications have been proposed [13, 10, 7, 14, 5, 12, 9]. The new OpenCL standard [1] should increase this research field and improve the portability of many-cores applications. However, lots of those references are "proof-of-concept" papers, and do not get actually used.

We present Biomanycores, a collection of many-cores bioinformatics tools, designed to bridge the gap between researches in high-performance-computing and usual bioinformaticians and biologists. The goal is both to gather many-cores programs and to propose interfaces to Bio* projects. The language of choice should be OpenCL, but, while no public implementation of OpenCL is available, CUDA projects are included.

The project is still in an early stage of development, but already includes 3 different applications: Smith-Waterman [10], pKnotsRG [11] and Position-Weight-Matrix scan [5], with interfaces to Biojava 1.6 [6], Bioperl 1.52 [15], and Biopython 1.50b [4]. We wish to open as much as possible Biomanycores to other high-performance bioinformatics applications and to better integrate to Bio* projects.

## References

[1] The Khronos Group, OpenCL 1.0 specification, 2008.

[2] Nvidia CUDA programming guide 2.0, 2008.

[3] M. Charalambous, P. Trancoso, and A. Stamatakis. Initial experiences porting a bioinformatics application to a graphics processor. *Adv. in Informatics*, pages 415–425, 2005.

[4] P. J. A. Cock, T. Antao, J. T. Chang, and al. Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics*, page btp163, 2009.

[5] M. Giraud and J.-S. Varré. Parallel position weight matrices algorithms. In *International Symposium on Parallel and Distributed Computing (ISPDC 2009)*, 2009.

[6] R. C. G. Holland, T. A. Down, M. Pocock, and al. BioJava: an open-source framework for bioinformatics. *Bioinformatics*, 24(18):2096–2097, 2008.

[7] L. Ligowski and W. Rudnicki. An efficient implementation of Smith-Waterman algorithm on GPU using CUDA, for massively parallel scanning of sequence databases. In *HiCOMB 2009*, 2009.

[8] W. Liu, B. Schmidt, G. Voss, and W. Müller-Wittig. GPU-ClustalW: using graphics hardware to accelerate multiple sequence alignment. In *High Performance Computing (HiPC 2006), LNCS 4297*, pages 363–374, 2006.

[9] Y. Liu, B. Schmidt, and D. Maskell. Parallel reconstruction of neighbor-joining trees for large multiple sequence alignments using CUDA. In *HiCOMB 2009*, 2009.

[10] S. A. Manavski and G. Valle. CUDA compatible GPU cards as efficient hardware accelerators for Smith-Waterman sequence alignment. *BMC Bioinformatics*, 9 Suppl 2:S10, 2008.

[11] J. Reeder, P. Steffen, and R. Giegerich. pknotsRG: RNA pseudoknot folding including near-optimal structures and sliding windows. *Nucl. Acids Res.*, 35(S2):W320–324, 2007.

[12] G. Rizk and D. Lavenier. GPU accelerated RNA folding algorithm. In *Using Emerging Parallel Architectures for Computational Science (ICCS 2009)*, 2009.

[13] M. C. Schatz, C. Trapnell, A. L. Delcher, and A. Varshney. High-throughput sequence alignment using graphics processing units. *BMC Bioinformatics*, 8:474, 2007.

[14] H. Shi, B. Schmidt, W. Liu, and W. Mueller-Wittig. Accelerating error correction in high-throughput short-read DNA sequencing data with CUDA. In *HiCOMB 2009*, 2009.

[15] J. E. Stajich, D. Block, K. Boulez, and al. The Bioperl toolkit: Perl modules for the life sciences. *Genome Research*, 12(10):1611–1618, 2002.