

# Accelerating Sequence Alignment

*Data-structures that map well are essential when porting applications to hybrid architectures*



Abdullah Gharaibeh



Matei Ripeanu

**T**his work sheds light on the importance of carefully considering the choice of the data structures used when porting applications to hybrid architectures (e.g., GPU-based platforms). In this context, we analyze and evaluate the impact of the data structures used to support ‘sequence alignment,’ one of the most computationally intensive bioinformatics problems. MUMmerGPU++, our exact sequence alignment tool that benefits from this analysis, demonstrates significant speedups compared to a state-of-the-art previous effort on real-life workloads (Figure 1).

## BACKGROUND

High-performance computing platforms are gradually shifting toward hybrid architectures that combine processors optimized for sequential

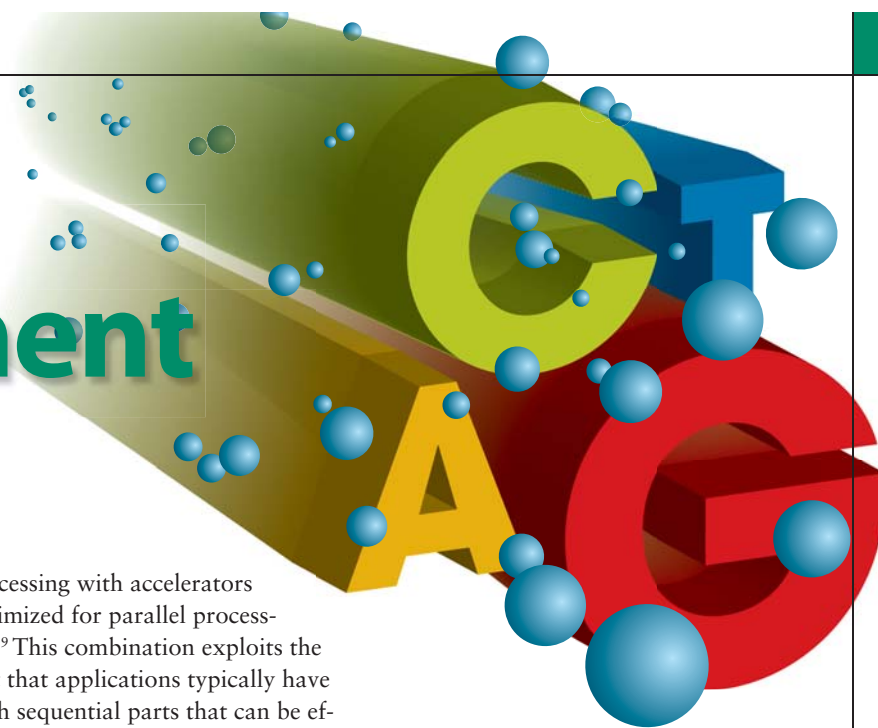
processing with accelerators optimized for parallel processing.<sup>9</sup> This combination exploits the fact that applications typically have both sequential parts that can be efficiently executed by the fast sequential processors, and parallel parts that are efficiently executed by massively-parallel accelerators. Overall, compared to homogeneous multicore systems, hybrid architectures offer a better balance between performance and used resources (i.e., power and processor area). Examples of such hybrid platforms include IBM’s Cell Broadband Engine, AMD’s Fusion architecture, Intel’s Larrabee chip, as well as commodity systems that host both traditional CPUs and commodity graphics processing units (GPUs).

Hybrid platforms that use GPUs have gained widespread popularity due to GPUs’ competitive pricing compared

to other accelerators, and their ability to deliver higher peak computational rate and memory bandwidth. Experience with these platforms includes reports of significant speedups compared to current homogeneous multicore systems in the same price range.<sup>1,2</sup> These reports ignited a passionate debate on the limits of GPU-supported acceleration for various classes of applications<sup>3,4</sup> and renewed interest in optimization techniques to harness this architecture.<sup>1,2</sup>

Indeed, designing applications to run efficiently on hybrid, GPU-based platforms is challenging for multiple reasons:

- First, GPUs offer a restricted form

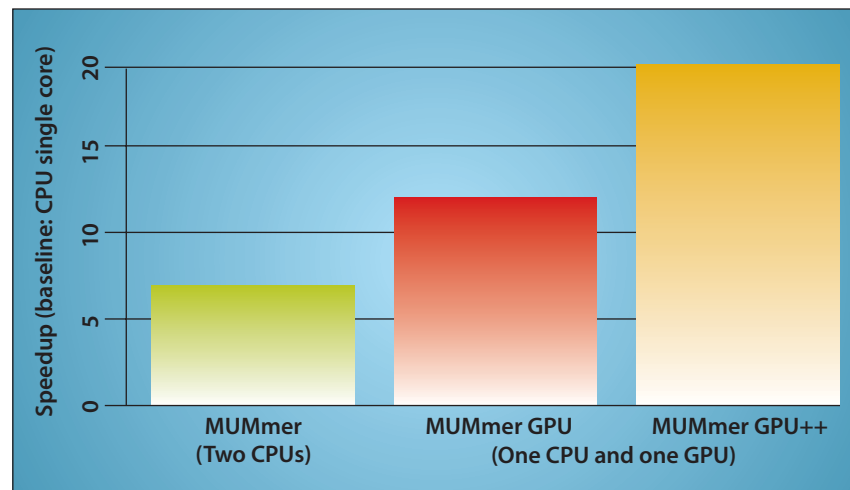


## HIGH PERFORMANCE COMPUTING

of parallelism, known as single-instruction multiple-data (SIMD), which allows for only one instruction to operate on multiple data items at each point in time. Hence, SIMD provides lower execution flexibility and requires extracting parallelism at the low-level.

■ Second, splitting the computation between the CPU and the GPU requires explicit data transfers between their address spaces over a shared I/O bus. Hence, efficiently scheduling data transfers between the two processing units, and finding a low coupling point that limits the volume of data transferred, are needed to achieve good performance.

■ Finally, the past experience on performance-efficient data structures needs to be carefully reconsidered when porting applications to use GPUs. The reason is that GPUs offer different computational tradeoffs compared to traditional multicore systems. On the one hand, GPUs offer one order of magnitude higher peak memory access bandwidth and peak computational power. On the other hand, current GPUs have limited, often an order of magnitude lower, internal memory space. Moreover GPUs' computational model results in extra overheads, as it relies on transferring data back and forth between the device and the system's main memory.



**Figure 1:** Plots the speedup of the three exact sequence alignment tools on two hardware platforms. The comparison baseline is the performance of MUMmer<sup>3</sup> running on a single CPU core (Intel E5540 @ 2.53 GHz). The first bar from the left presents the speedup offered by MUMmer when running on two high-end Intel quad core processors (E5540 @ 2.53 GHz) with 48 GB of system memory. The speedup is almost linear with the number of cores: 7x speedup for 8 cores. The workload was obtained from the NCBI trace archive ([www.ncbi.nlm.nih.gov](http://www.ncbi.nlm.nih.gov)) for a human genome with the following properties: average sequence length: ~200 nucleotides, number of sequences: ~10 million, reference sequence length: ~30million nucleotides (i.e., chromosome 21).

The second bar presents the speedup offered by MUMmerGPU,<sup>5,6</sup> a direct GPU-port of MUMmer, when executed on one quad core processor (E5520 @ 2.27GHz) and one GeForce GTX 480 GPU with 1.5GB of onboard memory on a system with 16GB of memory. MUMmerGPU offers a 12.27x speedup on this platform. Both MUMmer and MUMmerGPU use suffix trees as their core data structure.

The last bar presents the performance of our tool MUMmerGPU++<sup>7</sup> on the same platform as above. The reason for MUMmerGPU++'s impressive performance (over 20x speedup) is the use of a different data structure that better matches GPU characteristics: MUMmerGPU++ uses suffix arrays instead of suffix trees. Gharaibeh et al.<sup>7</sup> present evaluation results on different platforms and using other workloads.

## THE SEQUENCE ALIGNMENT PROBLEM

Sequence alignment (also known as 'read alignment') is a widely-used step in computational biology pipelines, such as comparative genomics, phylogenetic analysis and genome assembly. Sequence alignment aims to find all occurrences of each sequence of a large set of short sequences (called 'reads') in another, much longer sequence, called the 'reference sequence.' In this context, sequences are strings formed using the alphabet {A,C,G,T}.

Using a GPU to accelerate this operation is appealing for two reasons.

■ First, GPUs support higher peak memory bandwidth than traditional systems. This is facilitated by the faster memory technology used by GPUs, namely GDDR, and by employing a wider memory bus.

■ Second, parallelizing this operation is straightforward, since sequences can be processed independently, and the problem space can be easily partitioned. Therefore, it is not surprising that, after careful optimizations (e.g., to efficiently use GPUs' texture memory and improve data access locality) GPU ports of exact<sup>5,6</sup> or approximate<sup>10</sup> sequence alignment tools achieve significant speedups.

## HIGH PERFORMANCE COMPUTING

### OUR FINDINGS

We explored opportunities to further optimize the GPU ports of these tools: we aimed to explore the limits of potential gains brought by a careful space/time tradeoff analysis when porting applications to GPU-based hybrid platforms.<sup>7</sup> In particular, we analyzed these tradeoffs in the context of a well-engineered, widely-used ‘sequence alignment’ application<sup>8</sup> that has both CPU- and GPU-based implementations named MUMmer<sup>8</sup> and MUMmerGPU,<sup>5,6</sup> respectively.

Our main findings are as follows:


1. We confirm the feasibility of harnessing GPUs to accelerate sequence alignment. For our experimental setup (described in detail in Figure 1), the GPU-supported platform provides about 20x speedup compared to a single CPU core, and up to 3x speedup compared to multi-

threaded implementation running on two, high-end, Intel Xeon quad-core processors.

2. We demonstrate the importance of a careful choice of the data structure used to support GPU applications. We show that a data structure that matches well the space/time tradeoffs specific to a GPU can unlock dramatic performance gains. The direct implication of this observation is that, when porting applications to a hybrid, GPU-supported platform, designers should not only focus on extracting the application parallelism usable in a SIMD model; but, in order to maximize the performance gains, they may need to reconsider the choice of the data structures used.

3. We contrast, in the context of the sequence alignment application, the energy consumption of traditional and hybrid (GPU-enabled) systems. We show that, although the energy consumption rate of a hybrid system is higher, the total energy consumption to complete a full sequence alignment workload is lower due to its higher performance. While, for our experimental setup (see details in Figure 1), which compares hybrid and traditional systems, the hybrid system would require a performance gain of at least 37 percent to become more energy

efficient than the traditional one. Our implementation obtains higher speedups, thus making the hybrid system over 20 percent more energy efficient.

We have open-sourced MUMmerGPU++<sup>11</sup> the exact sequence alignment tool we designed and evaluated.<sup>7</sup> Our implementation, MUMmerGPU++, is compatible with the widely used MUMmer and MUMmerGPU tools. All of these tools also are part of NVIDIA’s bioinformatics benchmark suite.<sup>12</sup> 

### REFERENCES

1. D.B. Kirk and W.W. Hwu, *Programming Massively Parallel Processors: A Hands-on Approach*, Morgan Kaufmann, 2010.
2. W.W. Hwu, “GPU Computing Gems,” Morgan Kaufmann, 2011.
3. V. Lee, et al., *Debunking the 100X GPU vs. CPU myth: an evaluation of throughput computing on CPU and GPU*, ACM/IEEE International Symposium on Computer Architecture (ISCA 2010).
4. R. Vuduc, A. Chandramowlishwaran, J. Choi, M. Guney, A. Shringarpure, “On the Limits of GPU Acceleration,” Hot Topics in Parallelism (HotPar 2010).
5. M. C. Schatz, C. Trapnell, A. L. Delcher and A. Varshney, “High-throughput sequence alignment using Graphics Processing Units,” *BMC Bioinformatics*, vol. 8, pp. 474, Dec 10, 2007.
6. C. Trapnell and M. C. Schatz, “Optimizing data intensive GPGPU computations for DNA sequence alignment,” *Parallel Computing*, 2009.
7. A. Gharaibeh, M. Ripeanu, “Size Matters: Space/Time Tradeoffs to Improve GPGPU Applications Performance,” IEEE/ACM International Conference for High Performance Computing, Networking, Storage, and Analysis (SC 2010), New Orleans, LA, November 2010.
8. S. Kurtz, A. Phillippy, A. Delcher, M. Smoot, M. Shumway, C. Antonescu and S. Salzberg, “Versatile and open software for comparing large genomes,” *Genome Biol.*, vol. 5, pp. R12, 2004.
9. B. Catanzaro, A. Fox, K. Keutzer, D. Patterson, Su Bor-Yiing, M. Snir, K. Olukotun, P. Hanrahan, H. Chafi, “Ubiquitous Parallel Computing from Berkeley, Illinois, and Stanford,” *Micro, IEEE*, vol.30, no.2, pp.41-55, March-April 2010.
10. Liu, D. Maskell, and B. Schmidt. “CUDASW++: Optimizing Smith-Waterman Sequence Database Searches for CUDA-enabled Graphics Processing Units,” *BMC Research Notes*, 2(1):73, 2009.
11. MUMmerGPU++ is available at <http://netsyslab.ece.ubc.ca>
12. Available at: [http://www.nvidia.com/object/tesla\\_bio\\_workbench.html](http://www.nvidia.com/object/tesla_bio_workbench.html)

*Abdullah Gharaibeh is working toward his Ph.D., and Matei Ripeanu is an assistant professor with the Computer Engineering Department, at the University of British Columbia. They may be reached at [editor@ScientificComputing.com](mailto:editor@ScientificComputing.com)*