High-throughput Sequence Alignment Using Graphics Processing Units

Michael C. Schatz, Cole Trapnell, Arthur L. Delcher, and Amitabh Varshney

Center for Bioinformatics and Computation Biology, University of Maryland, College Park, MD, USA

http://mummergpu.sourceforge.net

Abstract

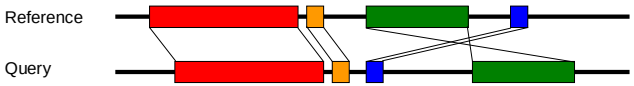
The recent availability of new, less expensive high-throughput DNA sequencing technologies has yielded a dramatic increase in the volume of sequence data that must be analyzed. These data are being generated for several purposes, including genotyping, genome resequencing, metagenomics, and de novo genome assembly projects. Sequence alignment programs such as MUMmer have proven essential for analysis of these data, but researchers will need ever faster, high-throughput alignment tools running on inexpensive hardware to keep up with new sequence technologies.

Traditionally, Graphics Processing Units (GPUs) have been highly specialized with two distinct classes of graphics stream processors: vertex processors, which compute geometric transformations on meshes, and fragment processors, which shade and illuminate the rasterized products of the vertex processors. Modern GPUs include several processors (tens to hundreds) of each type, and are organized in a streaming, data-parallel model in which the processors execute the same instructions on multiple data streams simultaneously. As GPUs have become increasingly more powerful and ubiquitous, though, researchers have begun using its power for non-graphics, or general-purpose (GPGPU) applications.

MUMmerGPU is a low cost, ultra-fast sequence alignment program designed to handle the increasing volume of data produced by new, high-throughput sequencing technologies. MUMmerGPU is a GPGPU drop-in replacement for MUMmer, using the GPUs in common workstations to simultaneously align multiple query sequences against a single reference sequence stored as a suffix tree. By processing the queries in parallel on the highly parallel graphics card, MUMmerGPU achieves more than a 10-fold speedup over a serial CPU version of the sequence alignment kernel, and outperforms MUMmer on a high end CPU by 3.5-fold in total application time when aligning reads from recent sequencing projects using Solexa/Illumina, 454, and Sanger sequencing technologies.

MUMmerGPU Algorithm

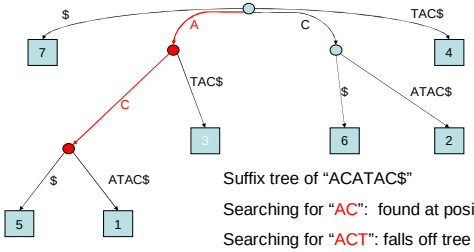
Sequence alignment algorithms find regions in one sequence, called the query sequence, that are similar or identical to regions in another sequence, called the reference sequence. MUMmerGPU, like its serial CPU counterpart MUMmer, aligns a set of query sequences against a reference sequence using a suffix tree and reports all exact alignments between the two. The exact alignments can be processed directly or used to seed longer in-exact alignments. Unlike its serial counterpart, the alignment kernel is executed in parallel on a highly parallel graphics card.



1. Construct Suffix Tree of Reference Sequence

A suffix tree is a tree which encodes every suffix of a sequence on a unique path from the root to a leaf. A sequence of length *n* has *n* suffixes and has *n* leaf nodes in the corresponding suffix tree. Edges are labeled with substrings of the reference sequence, and internal nodes represent positions where the suffixes diverge. Given a suffix tree, exact alignments between the reference and a query sequence can be found in time proportional to the length of the query by walking the tree from the root according to the characters in the query.

Suffix Tree Search



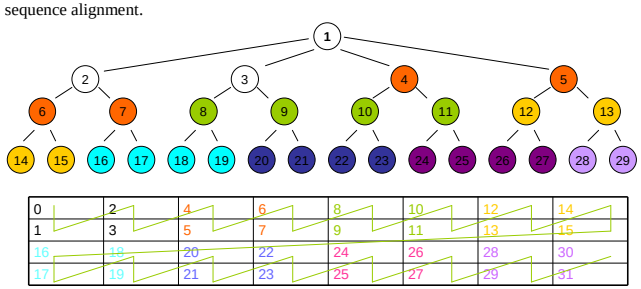
Suffix tree of "ACATAC$"

Searching for "AC": found at positions 1 & 5

Searching for "ACT": falls off tree => Not in Ref.

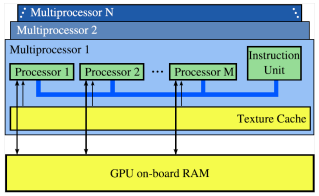
2. Optimize Tree Layout

MUMmerGPU uses nVidia G80 class graphics cards, such as the GTX 8800, which use a 2D cache for their on-board RAM. When the suffix tree is constructed, the nodes will be created with an arbitrary order, and scattered in RAM. MUMmerGPU therefore rearranges the nodes along a space filing curve into a 2D array so that a node and its children will be in close proximity in the graphics card's memory. This helps improve the cache hit rate during sequence alignment.



3. Transfer Data to GPU

The processors on the graphics card can read and write only to the on-board RAM. Therefore the suffix tree, and the query sequences are transferred in bulk to the GPU. The GTX 8800 has enough on-board RAM to store a tree for a several Mbp genome and tens of thousands of queries. If the reference and queries are too large to fit on the card, they are broken up into segments which are processed separately.



G80 Architecture

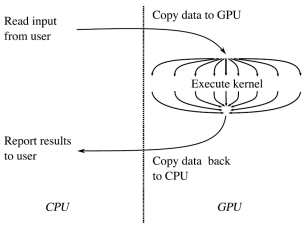
The GTX 8800 has 16 multiprocessors and 768 MB of ob-board RAM. Each multiprocessor has 8 processors, for a total of 128 processors running at 1.35 GHz. The 8 processors in a multiprocessor are controlled by a single instruction unit, and must execute the same instructions.

4. Align Sequences & Output Results

Once the suffix tree and query data are transferred to the GPU, MUMmerGPU executes the alignment kernel in parallel on the GPU. Each instance of the kernel runs on a single processor and executes the serial suffix tree alignment algorithm for a single query. The kernel finds all subsequences of the query greater than the minimal specified length (l) that exactly match the reference sequence. The alignment results are first written to the on-board RAM and then transferred back to the main system RAM after all of the alignments are complete. MUMmerGPU post-processes and prints the results on the CPU using the same output format as MUMmer.

Alignment Kernel

The alignment kernel was written in a restricted form of C using the Compute Unified Device Architecture (CUDA) from nVidia. CUDA makes it easy to compile and execute kernel code, but the GPU processors are limited and cannot use recursive functions or call stack.

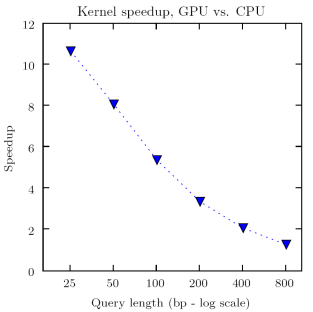


Results

We measured the performance of MUMmerGPU by comparing the execution time of the GPU and CPU version of the alignment code, and the total application runtime of MUMmerGPU versus MUMmer on a high end 3.0 GHz Intel Xeon with 2GB of system RAM. We ported MUMmerGPU to use the CPU instead of the GPU to isolate the benefit of using graphics hardware over running the same algorithm on the CPU. Porting MUMmerGPU to the CPU required only straightforward syntactic changes, and involved no algorithmic changes.

1. Synthetic Reads

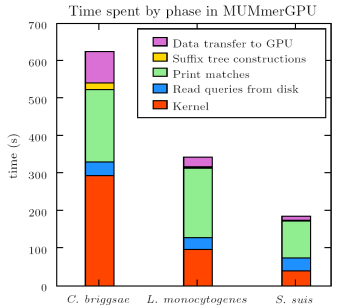
We aligned 50-, 100-, 200-, 400-, and 800-bp synthetically constructed reads to the *Bacillus anthracis* genome in order to explore MUMmerGPU's performance in the absence of errors and over a wider variety of query lengths then are available with genuine reads. Each test set contained exactly 250Mbp of query sequence divided evenly among all the reads in the set. For small query lengths, the GPU kernel executing in parallel was more than 10-fold faster than the CPU version of the kernel executed in serial. For longer reads, the speedup is less dramatic, due to the smaller cache size and decoherence of the GPU.



2. Genuine Reads

Next, we aligned reads from several sequencing projects against their genomes, as would be necessary for a resequencing or genotyping project.

Reference	Reference length (bp)	# of queries	Query length mean ± stdev	Min alignment length (l)	# of suffix trees (k)	Speedup
Caenorhabditis briggsae Chr. III Sanger sequencing	13,163,117	2,357,666	717.84 ± 159.44	100	2	3.71
Listeria monocytogenes 454 pyrosequencing	2,944,528	6,620,471	200.54 ± 60.51	20	1	3.79
Streptococcus suis Illumina/Solexa sequencing	2,007,491	26,592,500	35.96 ± 0.27	20	1	3.47



We aligned the reads against both strands of the chromosomal DNA for *L. monocytogenes* and *S. suis*, and against both strands of chromosome III of *C. briggsae*. In all cases we compared the end-to-end wall clock running time of MUMmerGPU versus MUMmer. Overall, MUMmerGPU was on average more than 3.5-fold faster execution running on the GPU than on the CPU. The running time of MUMmerGPU is dominated printing matches and other IO.

Conclusions

Our results show that a significant speedup, as much as a 10-fold speedup, can be achieved by executing the memory intensive sequence alignment program on the GPU with cached texture memory and data reordering. This speedup is realized only for large sets of short queries, but these read characteristics are beginning to dominate the marketplace for genome sequencing. For example Solexa- Illumina sequencing machines create on the order of 200 million 50bp reads in a single run. For a single human genotyping application, reads from a few runs need to be aligned against the entire human reference genome. Thus our application should perform extremely well on workloads commonly found in the near future.