GPUBwa -Parallelization of Burrows Wheeler Aligner using Graphical Processing Units

Abstract

A very popular discipline in bioinformatics is Next-Generation Sequencing (NGS) or DNA sequencing. It specifies sequencing methods for determining the order of nucleotide bases; adenine, guanine, cytosine and thymine in a molecule of DNA which is then assembled for analysis. A central challenge in DNA sequencing is sequence alignment, whereby fragments of much longer DNA sequence are aligned and merged in order to construct the original sequence. A wide variety of alignment algorithms have been subsequently developed over the past few years. Some commonly used softwares implementing these algorithms are BWA-SW, Bowtie, Velvet, SOAP and MAQ. Most of these take lot of time to execute on general purpose processors. Hardware accelerators such GPUs and FPGAs can be used with processors to fasten these applications. As per our knowledge there are very few reports published in literature about accelerating these applications using GPUs. In this report, we describe the implementation of GPUBwa, a GPU sequence alignment software that is based on BWA, to accelerate the alignment of sequencing reads generated by these instruments to a reference DNA sequence.

Background and Related work

Next-generation sequencing (NGS) is a technique based on sequencing by synthesis or sequencing by ligation in a massively parallel fashion and generates short sequencing reads ranging from 25 to 400 bp. The first commercially available next-generation sequencer, the Genome Sequencer 20 was released by 454 Life Sciences in 2005 with the hope of enabling the analysis of complete genomes within a short period of time. It produced a throughput of 20 megabases from a 5-hour run, which was 30 fold higher than traditional Sanger capillary electrophoresis systems. Over these years, the output of next-generation sequencers has increased 30, 000 fold to 600 gigabases from a single instrument run (Illumina Hiseq 2000) which is about 200 times the depth of coverage of an entire human genome.

The advancement of the technology has generated an enormous amount of sequence data. Sequence alignment is one of the first steps for downstream data analyses, during which sequencing reads have to be mapped either to other reads to form a genome (also known as de novo sequence assembly); or on to a reference DNA sequence, usually a genome, for downstream applications such as single-nucleotide polymorphism (SNP) discovery, Chip-Seq or RNA-Seq. Here we focus on the latter type of alignments.

A handful of software packages have been developed for computing alignments of sequencing reads generated by NGS instruments on to a reference sequence. Early generation aligners such as MAQ, RMAP and Soap use hash-based algorithms to perform the mapping of sequencing reads on to reference genomes. Even though these tools can be accelerated by several seeding approaches, or parallelized by using multiple cores and cloud computing (e.g. CloudBurst), the computational cost remains expensive. For example, by extrapolating the result from Schtaz et al., it would take over 30, 000 CPU hours to align reads generated from a single HiSeq 2000 run.

Later in 2009, a new generation of sequence aligners were released, namely Soap2, Bowtie and BWA. These tools use a suffix tree-based algorithm (also known as FM-index) based on Burrows-Wheeler Transform (BWT). BWT is a block-sorting algorithm originally designed for lossless data compression, it can also be used for string matching by a backward search approach. The major advantages of this approach include a low time complexity of O(n) to find an exact match where n is the length of the query and the performance is independent from the size of the reference sequence. In addition, a high compression ratio of 2.5 bit per base for the reference genome also means a full human genome can fit into 1.3 GB of space. The new algorithm is a magnitude quicker and much more memory efficient than their hash-based predecessors. In an experiment performed by Li and Durbin the alignment throughput for 12.2 million 51 bp reads being mapped to the Human genome went down from 94 CPU hours (MAQ) to 4 CPU hours (BWA) on a 2.5 GHz Intel Harpertown-based Xeon E5420 while retaining comparable accuracy in alignment mapping.

Modern graphics cards are designed primarily for rendering real time, high-definition complex 3D graphics features for various visual applications such as gaming and graphics design. Each graphics processing unit, or GPU, consists of many high performance stream processors capable of performing billions of independent calculations per second in order to meet the high visualization demand required for graphics applications. It is this processing capability that can be translated into a general-purpose computation capability equivalent to a small cluster of traditional CPUs. In addition, the lower energy profile and cost means the use of GPU to perform parallel computing tasks has become increasingly attractive. Many modern supercomputers including the Chinese Tianhe-1A, Nebulae and Japanese Tsubame 2.0 also contain multiple GPU nodes on top of traditional nodes with CPUs to take advantage to the parallel computing capability of GPUs.

MUMmerGPUis one of the first GPGPU-based DNA alignment software that utilizes the NVIDIA Compute Unified Device Architecture (CUDA) to perform variable-length maximal exact DNA alignments. Unlike other BWT aligners it uses a different suffix-tree approach, namely Ukkonen's algorithm to find exact matches in the reference sequence of all the sub-strings in the query DNA sequence. The current version of MUMmer GPU out-performs its CPU counterpart by 13 fold. However, unlike other sequence aligners mentioned in the previous section, MUMmerGPU does not support inexact alignments by itself and has to be used in conjunction with the original MUMmer software package to perform inexact alignments.

Here, we present GPUBwa, a program which aims to speed up the BWA software tool which is based on the BWT-based alignment core module to make use of the massive parallelism of GPU. It also employs the fast and memory efficient BWT-based algorithm employed in the original software and supports mismatches and full gapped alignment of sequencing reads.

Profiling and selection of BWA algorithm for implementation

Results of Profiling of some of the widely used algorithms was done as shown in table 1.

Chromosome	Size of the reference genome(MB)	Algorithm	No. of bp in the query sequence	Time taken to index(s)	Time taken to align(s)
10	131.8	maq	70	40.4	741.6
		soap	70	194.1	79.4
		bwa*	70	232.4	315.2
		bwa-sw*	500	232.4	8838.5
19	57.5	maq	70	31.5	865.2
		soap	70	78.2	84.2
		bwa	70	92.7	359.4
		bwa-sw	500	92.7	8030.5
X	151	maq	70	41.1	802.7
		soap	70	225.8	86.4
		bwa	70	267.5	342.6
		bwa-sw	400	267.5	11269.8

Table 1: Profiling results of some of the most widely and highly cited algorithms till date on 3 different human chromosomes. The profiling was done with Intel V-tune Profiler.

We chose BWA for implementation due to the following reasons

1. **Potential areas for speedup:** The function graph of BWA is shown in table 2. As can be seen from the table, The bwa_aln_core function consists the critical section of the running time and thus is a potential area for speedup. It is also to be noted that this function only consists of 5-10% of the entire code of BWA but shares a very high portion of the running time. These are potential areas for loop unrolling and implementation on GPU
2. **External validity for hypothesis:** BWT algorithm, which forms the heart of BWA algorithm has already been implemented on GPU[3] and showed significant improvement in time.
3. **Popular:** BWA is a highly popular and widely used alignment algorithm. Speeding it up would result in speeding up of computer based biological sequence alignment which will have a great impact on entire chain of its applications in bioinformatics, forensics etc.

Algorithm	Function	% Contrib	Remarks
bwa-index	bwa_index	100	It calls a lot of functions each consuming a significant amount of time.
bwa	bwa_cal_sa_reg_gap	99.1	bwa_cal_sa_reg_gap is a sub-function of
	bwa_aln_core	100	bwa_aln_core which constitutes a critical part of it
bwa-sw	bsw2_aln_core	99.6	The functions above it constitute a lot of code but take very less time.

Table 2: Function graph of BWA algorithm

BWA algorithm[2] constructs a suffix array interval (BWT) from ref gnome and aligns short query sequence to it. Initial construction of suffix array takes large time linear in length of query sequence. The alignment time is linear w.r.t the length of query sequence using backward search and independent of the size of the ref. gnome.

Implementation

We mapped the code to GPU to parallelize the alignment of sequence reads. Firstly, the program loads the complete BWT-encoded reference sequence and sequence reads from disk to GPU memory; This is followed by launching a GPU alignment kernel, where the alignment task of each of the sequence reads are distributed to hundreds of processors within the GPU and computations are performed in parallel; Once the kernel finishes, the alignment results are transferred from GPU back to disk. The following sections describe the details of each of the steps performed:

1. Transferring BWT-encoded reference sequence and sequence reads from disk to GPU

The code first loads the full BWT suffix array from disk into cached texture memory in the GPU using a 1-dimensional array.

2. CUDA thread assignments

Mapping a sequence read to a reference sequence is a data independent process and does not require any information from any of the other reads, thus we employ a straightforward data parallelism by assigning an alignment kernel thread to each of the individual sequencing reads and launching the GPU kernel with thousands of threads at the same time.

3. Inexact sequence alignment--a depth-first search GPU kernel

The alignment in BWA consists of a backward string matching algorithm. It uses an efficient breadth-first search (BFS) approach and can utilize a lot of space for each thread. With thousands of concurrent threads on the GPU, the memory to each thread is very limited and BFS does not seem to be an option. Therefore, we to adopt modified DFS approach to perform alignments.

4. Copying the results back to disk for output

The alignment results are copied back from device to the disk and converted to .sai format on the host for output

Results

1. The accuracy of GPUbwa is similar to Bwa

The changes in the search algorithm and alignment criteria in GPUBwa do not penalize the alignment accuracy of the software. We tested the accuracy of both the tools using 1 million reads of 70bp generated from wgsim tool[1] for D. Melanogaster and E. Clegane genome. Table 3 shows the comparison of the mapping percentage and the error in alignment. Both the mapping percentage and the error in alignment are similar for GPUBwa and Bwa confirming the correctness of the algorithm.

	BWA	**GPUbwa**
Map percentage	89%	88%
Error	0.05%	0.05%

Table 3(a): Accuracy comparison of BWA and GPUBwa for random generated short reads (70 bp) of *D. Melanogaster* via wgsim

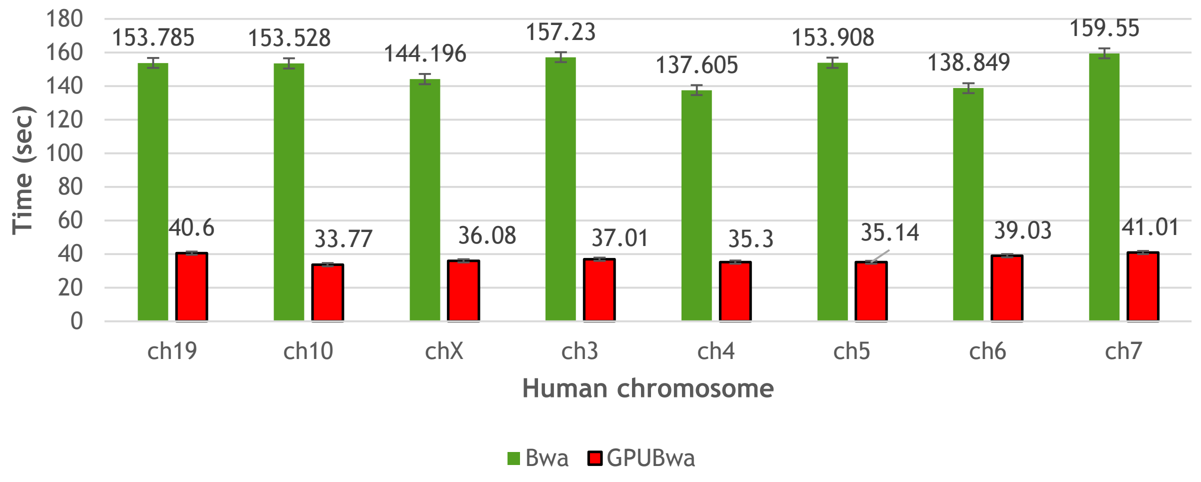
	BWA	**GPUbwa**
Map percentage	89.95%	90.01%
Error	0.06%	0.04%

Table 3(b): Accuracy comparison of BWA and GPUBwa for random generated short reads (70 bp) of *C. Elegans* via wgsim

2. GPUBwa performs faster than BWA

Comparison of alignment times for alignment of random reads (70bp) of human chromosomes for both GPUBwa and Bwa are shown in figure 1. The reduction in time was highly significant *(pairise t-test, two tallied p value<0.0001)*. The average speedup obtained over 6 core multi-threaded implementation of BWA v/s 2 core GPU implementation of GPUBwa was approx. 4.04. This showed that there is a great potential for improvement over multi-GPU systems.

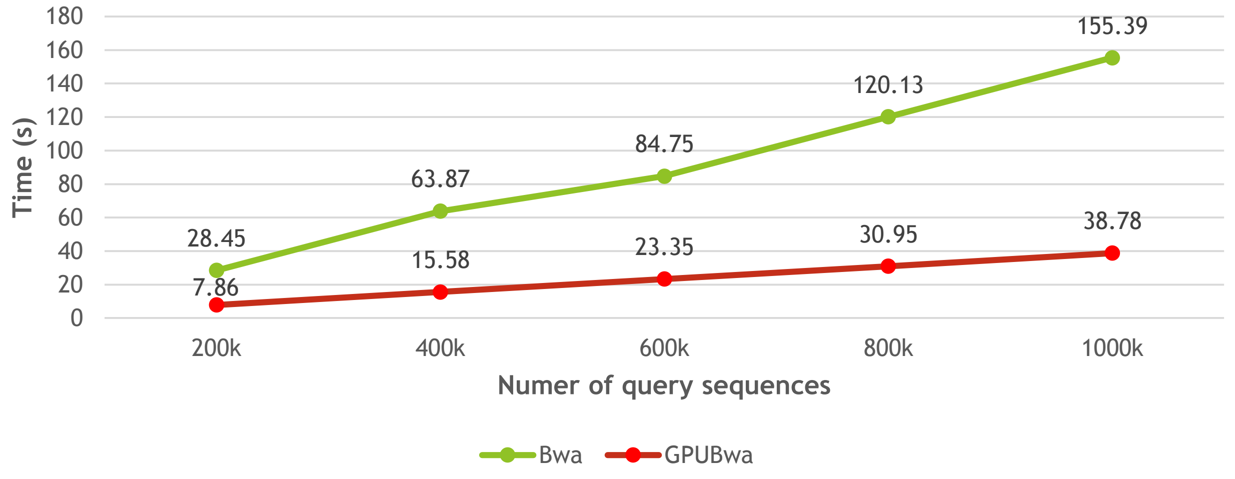
Figure 1: Comparison of time taken for alignment with Bwa and GPUBwa for 1 million query sequences. Specs: Intel(R) Xeon(R) CPU X5650 @ 2.67GHz (6 cores)
nVidia Corporation GF100 [Tesla M2070] (rev a3) (2 cores)



3. GPUBwa scales better than BWA with the increasing number of query sequences.

As the number of sequences for the query sequences increase, GPUBwa shows a better alignment time growth as compared to BWA. We tested the running time of the both the tools for different number of randomly query sequences (70bp each) of Human Chromosome 19. This comparison is shown in figure 2. It is useful to note that the difference between the runtime of GPUBWA and BWA increases with the amount of query sequences (on positive side for GPUBwa) which shows that GPUBWA scales better than BWA for large number of query sequence. Please note that this is subject to the fact that the entire array of query sequences should fit in the GPU. Our implementation as of yet does not handle split implementations.

Figure 2: Comparison of align times for Chromosome 19 ref. gnome with different number of query sequences



Further Scope

1. Implementation of BWA constructing indexing algorithm on GPU

BWA algorithm employs three steps for the sequence alignment. 1) Indexing to contrast BWT suffix array from the ref. gnome. 2) Alignment using backward search and 3) Converting output to human readable SAM format. We have only implemented the backward search alignment on GPU. We are using the same indexing and output conversion algorithm as BWA. As a future work, the indexing algorithm can also be implemented on GPU for additional speedup since as literature suggest BWT algorithm shows considerable speedup when implemented on GPU[3].

2. Analysis of scalability with multiple GPU systems

We only had system having 2 GPUs at our disposal. As a result we weren't able to test the code over multiple GPUs. Future work involves testing the scalability of the program on the multiple GPU systems. We hypothesize that considerable speedup will be obtained on multiple GPU systems.

References

[1] Heng Li. *Wgsim, Samtools software package*. Available. [Online] https://github.com/lh3/wgsim

[2] Li H. and Durbin R. (2010) Fast and accurate long-read alignment with Burrows-Wheeler Transform. Bioinformatics, Epub.

[3] Drozd, Aleksandr, Naoya Maruyama, and Satoshi Matsuoka. "Fast GPU Read Alignment with Burrows Wheeler Transform Based Index." *Proceedings of the 2011 ACM/IEEE conference on Supercomputing (SC'11)*. 2011.

(Dhruv Jain and Amit Jaiswal)