Fine-Grain Parallel Megabase Sequence Comparison with Multiple Heterogeneous GPUs

Edans F. de O. Sandes
University of Brasilia
edans@cic.unb.br

Guillermo Miranda
Barcelona Supercomputing Center
guillermo.miranda@bsc.es

Alba C. M. A. Melo
University of Brasilia
alba@cic.unb.br

Xavier Martorell
Universitat Politècnica de Catalunya
Barcelona Supercomputing Center
xavier.martorell@bsc.es

Eduard Ayguadé
Universitat Politècnica de Catalunya
Barcelona Supercomputing Center
eduard.ayguade@bsc.es

Abstract

This paper proposes and evaluates a parallel strategy to execute the exact Smith-Waterman (SW) algorithm for megabase DNA sequences in heterogeneous multi-GPU platforms. In our strategy, the computation of a single huge SW matrix is spread over multiple GPUs, which communicate border elements to the neighbour, using a circular buffer mechanism that hides the communication overhead. We compared 4 pairs of human-chimpanzee homologous chromosomes using 2 different GPU environments, obtaining a performance of up to 140.36 GCUPS (Billion of cells processed per second) with 3 heterogeneous GPUS.

Categories and Subject Descriptors D.1.3 [*Programming Techniques*]: Concurrent Programming; J.3 [*Life and Medical Sciences*]: Biology and Genetics

Keywords GPU; Biological Sequence Comparison; Smith-Waterman;

1. Introduction

Smith-Waterman (SW) [4] is an exact algorithm based on the longest common subsequence (LCS) concept, that uses dynamic programming to find local alignments between two sequences. SW is very accurate but it needs a lot of computational resources. GPUs (Graphics Processing Units) have been considered to accelerate SW, but very few GPU strategies [1, 3] allow the comparison of Megabase sequences longer than 10 Million Base Pairs (MBP). SW#[1] uses 2 GPUs to execute a Myers-Miller [2] linear space variant of SW. CUDAlign [3] uses a single GPU to execute a combined strategy with SW and Myers-Miller. When compared to SW#(1 GPU), CUDAlign (1 GPU) presents better execution times for huge sequences [1].

In this work, we modified the most computational intensive stage of CUDAlign, parallelizing the computation of a single huge DP matrix among heterogeneous GPUs in a fine-grained way. In the proposed strategy, GPUs are logically arranged in a linear way so that each GPU calculates a subset of columns of the SW matrix, sending the border column elements to the next GPU. Experimental results collected in 2 different environments show performance of up to 140 GCUPS (Billion of cells processed per second) using 3 heterogeneous GPUS. With this performance, we are able to compare real megabase sequences in reasonable time.

2. Proposed Multi-GPU Strategy

We modified the first stage of CUDAlign [3] to parallelize computation of a single huge DP matrix among many heterogeneous GPUs. The parallelization is done using a multi-GPU wavefront method, where the GPUs are logically arranged in a linear way, i.e, the first GPU is connected to the second, the second to the third and so on. Each GPU computes a range of columns of the DP matrix and the GPUs transfer the cells of their last column to the next GPU. In a scenario composed of heterogeneous GPUs, assigning the same number of columns to all GPUs is not a good choice. In this case, the slowest GPU would determine the processing rate of the whole wavefront. To avoid this, we statically distribute the columns proportionally to the computational power of each GPU. This distribution can be obtained from sequence comparison benchmarks that determine each GPU

PPoPP '14, February 15–19, 2014, Orlando, Florida, USA.
Copyright is held by the owner/author(s).
ACM 978-1-4503-2656-8/14/02.
http://dx.doi.org/10.1145/2555243.2555280

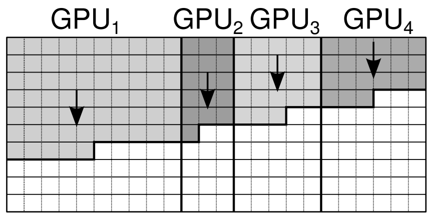

Figure 1. Columns distributions for 4 GPUs.

Chr.	Human		Chimpanzee		Score
	Accession	Size	Accession	Size	
chr19	NC_000019.9	59M	NC_006486.3	64M	17297608
chr20	NC_000020.10	63M	NC_006487.3	62M	40050427
chr21	NC_000021.8	48M	NC_006488.2	46M	36006054
chr22	NC_000022.10	51M	NC_006489.3	50M	31510791

Table 1. Sequences used in the tests.

computational power, in cells updated per second (CUPS). Then, the column distribution is defined proportionally to the computational power obtained in the comparisons. Figure 1 shows a possible column distribution for a 4-GPU configuration, where the faster GPUs are responsible to process more columns than the slower GPUs.

In our strategy, each GPU is bound to one process and each process has 3 CPU threads: one manager thread, that manages computation in GPU, and two communication threads, that handle inter-process communication overlapped with the computation. The inter-process communication is made by circular buffers and sockets, which transfer cells from one process to the next. In hosts with multiple attached GPUs, one process is created for each GPU, and the connection is handled by loopback sockets. In environments with different hosts, the connections are made using TCP sockets.

3. Experimental Results

We compared human and chimpanzee chromosomes 19 to 22, which sizes vary from 46MBP (Million Base Pairs) to 64MBP. As far as we know, this was the first time chromosomes 19, 20 and 22 were compared with SW. For validation purposes, the accession numbers, sizes and the optimal local scores obtained during our tests are presented in Table 1. The SW score parameters used in the tests were: match: $+1$; mismatch -3; first gap: -5; extension gap: -2.

The multi-GPU version of CUDAlign was tested in two heterogeneous multi-GPU environments: Panoramix and Laico. **Panoramix** is a single host connected to 1 Tesla K20c and 2 Tesla C2050. The column distribution was 46.10% to the K20c GPU and 26.95% to each C2050 GPU. **Laico** (LAboratory of Integrated and COncurrent systems) has several hosts with GPUs. For our tests, we selected one host with a GTX 580 GPU and two hosts with a GTX 680 GPU each. The column distribution was 30.71% for the GTX 580 and 34.64% for each GTX 680.

Table 2 presents execution times and GCUPS for the comparisons in both heterogeneous environments. It shows that the relative performance ranged from 100.62 to 101.38 GCUPS (Billion of cells processed per second) in Panoramix and from 139.60 to 140.36 GCUPS in Laico. We can see that the results are very uniform, since the GCUPS varied in less than 0.76% in Panoramix and 0.55% in Laico. By dividing the GCUPS by the defined columns proportion in Panoramix, the K20c GPU was responsible for approximately 46 GCUPS and each C2050 GPU was responsible for approximately 27 GCUPS. For Laico, the GTX 580 was responsible for approximately 43 GCUPS and each the GTX 680 was responsible for 48 GCUPS.

Chr.	Size	Panoramix		Laico	
		Time	GCUPS	Time	GCUPS
chr19	$59M\times64M$	37252s	101.02	26957s	139.60
chr20	$63M\times62M$	38537s	100.96	27728s	140.31
chr21	$48M\times46M$	22238s	100.62	16024s	139.63
chr22	$51M\times50M$	25416s	101.38	18180s	140.36

Table 2. Execution Times and GCUPS.

4. Future Works

As future work, we intend to execute our parallel strategy in a dedicated cluster environment with several GPUs. In order to use the cluster resources in a reasonable way, we will develop a formal method to predict the execution time and speedup of a comparison given the size of the sequences and the number of GPUs.

Acknowledgments

We thankfully acknowledge the support of the grant SEV-2011-00067 of Severo Ochoa Program, awarded by the Spanish Government, the Spanish Ministry of Science and Technology (TIN2012-34557, CSD2007-00050) and the Generalitat de Catalunya (2009-SGR-980). This work is also partially supported by CNPq/Brazil (grants 242800/2012-2 and 211456/2013-6) and CAPES/PVE (grant 02/2012).

References

[1] M. Korpar and M. Sikic. SW#-GPU-Enabled Exact Alignments on Genome Scale. *Bioinformatics*, 29(19):2494–2495, 2013.

[2] E. W. Myers and W. Miller. Optimal Alignments in Linear Space. *Computer Applications in the Biosciences*, 4(1):11–17, 1988.

[3] E. Sandes and A. de Melo. Retrieving Smith-Waterman Alignments with Optimizations for Megabase Biological Sequences Using GPU. *IEEE Transactions on Parallel and Distributed Systems*, 24(5):1009–1021, 2013.

[4] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *J Mol Biol*, 147(1):195–197, 1981.