Algorithmic Improvement and GPU Acceleration of the GenASM Algorithm

Joël Lindegger
ETH Zürich
lijoel@ethz.ch

Damla Senol Cali
Bionano Genomics
damlasenolcali@gmail.com

Mohammed Alser
ETH Zürich
alserm@ethz.ch

Juan Gómez-Luna
ETH Zürich
el1goluj@gmail.com

Onur Mutlu
ETH Zürich
omutlu@gmail.com

***Abstract*—We improve on GenASM, a recent algorithm for genomic sequence alignment, by significantly reducing its memory footprint and bandwidth requirement. Our algorithmic improvements reduce the memory footprint by 24× and the number of memory accesses by 12×. We efficiently parallelize the algorithm for GPUs, achieving a 4.1× speedup over a CPU implementation of the same algorithm, a 62× speedup over minimap2's CPU-based KSW2 and a 7.2× speedup over the CPU-based Edlib for long reads.**

***Index Terms*—read mapping, sequence alignment, GPU, memory**

I. Introduction

Efficient algorithms for solving the genomic sequence alignment problem are based on *dynamic programming (DP)*, such as the Smith-Waterman-Gotoh algorithm [1]. These algorithms first construct a *DP table* to find the best possible alignment score, followed by a *traceback* step to retrieve the optimal alignment. They have quadratic time and space complexity [2], and asymptotically faster solutions are not to be expected [3]. Hence, a significant effort has been and is being put towards speeding up this step through several approaches, such as prior filtering (e.g. [4–7]), constant factor algorithmic speedups (e.g. [8–10]), GPU-based acceleration (e.g. [11, 12]), FPGA-based acceleration (e.g. [13]) or through specialized hardware accelerators (e.g. [14–16]).

Our **goal** is to speed up genomic sequence alignment over state-of-the-art software solutions for both short and long reads. To this end, we develop novel algorithmic improvements for the GenASM algorithm [14] and accelerate it using a GPU.

We choose the GenASM algorithm for its high throughput and fine-grained parallelism that makes it well suited for a GPU implementation. We observe that the GenASM algorithm exhibits high memory bandwidth pressure and its working set (DP table) does not fit into on-chip memory. We alleviate these limitations with three **key ideas**: We discover that (1) the DP table can be *compressed* by storing only the bitwise AND of the variables for each DP entry, (2) part of the DP table can opportunistically be *excluded from calculation* if previous rows of the DP table already contain the full solution, and (3) part of the DP table *does not need to be stored* because the traceback operation cannot reach these entries. As a result, the *entire* DP table fits into fast on-chip memory, and the number of accesses to the DP table is reduced significantly.

The **contributions** of this work are as follows:

- We develop three novel algorithmic improvements to GenASM, collectively reducing the memory footprint by 24× and the number of memory accesses by 12×.
- Based on these insights, we develop CPU and GPU implementations of our improved GenASM algorithm that are capable of aligning both short and long reads.
- We demonstrate that our CPU and GPU implementations provide large speedups over the state-of-the-art sequence alignment software, KSW2 [9, 10] and Edlib [8].
- We demonstrate that our algorithmic improvements are effective, and our GPU implementation efficiently parallelizes the improved GenASM algorithm.

II. Results

We evaluate the CPU implementation of our improved GenASM algorithm and the baseline sequence aligners (i.e. KSW2 [9, 10] and Edlib [8]) on a dual socket Intel Xeon Gold 5118 (2× 24 logical cores) at 3.2GHz with 196GB DDR4 RAM, using 48 threads. We run our GPU implementation on an NVIDIA A6000 [17]. We simulate 500 PacBio reads from the human genome using PBSIM2 [18], each of length 10 kb. We map these reads to the human genome using minimap2 [10] and obtain all chains (candidate locations) it generates using the `-P` flag, 138,929 locations in total. We align the (read, reference) pairs obtained from the candidate locations using KSW2 [9, 10], Edlib [8], and both our CPU and GPU implementations.

Our CPU implementation achieves a 15.2×, 1.7×, and 1.9× speedup over KSW2, Edlib, and a CPU implementation of GenASM without our improvements, respectively. Our GPU implementation achieves a 4.1×, 62×, 7.2×, and 5.9× speedup over our CPU implementation, KSW2, Edlib, and and a GPU implementation of GenASM without our improvements, respectively. We observe that the CPU and GPU implementations of GenASM provide speedups over Edlib *only* if our algorithmic improvements are applied. We conclude that our algorithmic improvements are effective, and enable CPU and GPU implementations of GenASM to outperform state-of-the-art genomic sequence aligners.

References

[1] O. Gotoh, "An Improved Algorithm for Matching Biological Sequences," *Journal of Molecular Biology*, 1982.
[2] M. Alser *et al.*, "Technology dictates algorithms: Recent developments in read alignment," *Genome Biology*, 2021.
[3] A. Backurs *et al.*, "Edit Distance Cannot Be Computed in Strongly Subquadratic Time (Unless SETH is False)," *STOC*, 2015.
[4] H. Xin *et al.*, "Shifted Hamming distance: a fast and accurate SIMD-friendly filter to accelerate alignment verification in read mapping," *Bioinformatics*, 2015.
[5] M. Alser *et al.*, "SneakySnake: a fast and accurate universal genome pre-alignment filter for CPUs, GPUs and FPGAs," *Bioinformatics*, 2020.
[6] G. Singh *et al.*, "FPGA-Based Near-Memory Acceleration of Modern Data-Intensive Applications," *IEEE Micro*, 2021.
[7] N. Mansouri Ghiasi *et al.*, "GenStore: a high-performance in-storage processing system for genome sequence analysis," *ASPLOS*, 2022.
[8] M. Šošić *et al.*, "Edlib: a C/C++ library for fast, exact sequence alignment using edit distance," *Bioinformatics*, 2017.
[9] H. Suzuki *et al.*, "Introducing difference recurrence relations for faster semi-global alignment of long sequences," *BMC Bioinformatics*, 2018.
[10] H. Li, "Minimap2: pairwise alignment for nucleotide sequences," *Bioinformatics*, 2018.
[11] N. Ahmed *et al.*, "GASAL2: a GPU accelerated sequence alignment library for high-throughput NGS data," *BMC Bioinformatics*, 2019.
[12] N. Ahmed *et al.*, "GPU acceleration of Darwin read overlapper for de novo assembly of long DNA reads," *BMC Bioinformatics*, 2020.
[13] X. Fei *et al.*, "FPGASW: accelerating large-scale Smith–Waterman sequence alignment application with backtracking on FPGA linear systolic array," *Interdiscip Sci*, 2018.
[14] D. Senol Cali *et al.*, "GenASM: A high-performance, low-power approximate string matching acceleration framework for genome sequence analysis," *MICRO*, 2020.
[15] Y. Turakhia *et al.*, "Darwin: A Genomics Coprocessor," *IEEE Micro*, 2019.
[16] M. Alser *et al.*, "Accelerating genome analysis: a primer on an ongoing journey," *IEEE Micro*, 2020.
[17] NVIDIA, "NVIDIA RTX A6000 Graphics Card," https://www.nvidia.com/en-us/design-visualization/rtx-a6000, 2020.
[18] Y. Ono *et al.*, "PBSIM2: a simulator for long-read sequencers with a novel generative model of quality scores," *Bioinformatics*, 2020.