GenASM: A High-Performance, Low-Power Approximate String Matching Acceleration Framework for Genome Sequence Analysis

Damla Senol Cali
Carnegie Mellon University
(dsenol@andrew.cmu.edu)

Gurpreet S. Kalsi\textsuperscript{2}, Zulal Bingol\textsuperscript{3}, Can Firtina\textsuperscript{4}, Lavanya Subramanian\textsuperscript{5}, Jeremie Kim\textsuperscript{1,4}, Rachata Ausavarungnirun\textsuperscript{6,1}, Mohammed Alser\textsuperscript{4}, Juan Gomez-Luna\textsuperscript{4}, Amirali Boroumand\textsuperscript{1}, Anant Nori\textsuperscript{2}, Allison Scibisz\textsuperscript{1}, Sreenivas Subramoney\textsuperscript{2}, Can Alkan\textsuperscript{3}, Saugata Ghose\textsuperscript{7,1}, and Onur Mutlu\textsuperscript{4,1,3}
Genome Sequencing

- **Genome sequencing**: Enables us to determine the order of the DNA sequence in an organism’s genome
  - Plays a **pivotal role** in:
    - Personalized medicine
    - Outbreak tracing
    - Understanding of evolution

- Modern genome sequencing machines extract smaller randomized fragments of the original DNA sequence, known as **reads**
  - **Short reads**: a few hundred base pairs, error rate of $\sim 0.1\%$
  - **Long reads**: thousands to millions of base pairs, error rate of 10–15\%
Genome Sequence Analysis

- **Read mapping**: *First key step* in genome sequence analysis (GSA)
  - Aligns *reads* to one or more possible locations within the *reference genome*, and
  - Finds the *matches* and *differences* between the read and the reference genome segment at that location

- Multiple steps of read mapping require *approximate string matching*
  - Approximate string matching (ASM) enables read mapping to account for *sequencing errors* and *genetic variations* in the reads

- Bottlenecked by the *computational power and memory bandwidth limitations* of existing systems
GenASM: ASM Framework for GSA

Our Goal:

Accelerate approximate string matching by designing a fast and flexible framework, which can accelerate multiple steps of genome sequence analysis.

- **GenASM:** First ASM acceleration framework for GSA
  - Based upon the *Bitap* algorithm
    - Uses fast and simple bitwise operations to perform ASM
  - Modified and extended ASM algorithm
    - Highly-parallel *Bitap* with long read support
    - Bitvector-based novel algorithm to perform *traceback*
  - Co-design of our modified scalable and memory-efficient algorithms with low-power and area-efficient hardware accelerators
GenASM: Hardware Design

**GenASM-DC:** generates bitvectors and performs edit Distance Calculation

**GenASM-TB:** performs TraceBack and assembles the optimal alignment
GenASM: Hardware Design

Our specialized compute units and on-chip SRAMs help us to:

→ Match the rate of computation with memory capacity and bandwidth

→ Achieve high performance and power efficiency

→ Scale linearly in performance with the number of parallel compute units that we add to the system
GenASM-DC: Hardware Design

- **Linear cyclic systolic array** based accelerator
  - Designed to **maximize parallelism** and **minimize memory bandwidth and memory footprint**
Very simple logic:

1. Reads the bitvectors from one of the TB-SRAMs using the computed address
2. Performs the required bitwise comparisons to find the traceback output for the current position
3. Computes the next TB-SRAM address to read the new set of bitvectors
Key Results – Area and Power

- Based on our synthesis of GenASM-DC and GenASM-TB accelerator datapaths using the Synopsys Design Compiler with a 28nm LP process:
  - Both GenASM-DC and GenASM-TB operate @ 1GHz

<table>
<thead>
<tr>
<th>Component</th>
<th>Area (mm²)</th>
<th>Power (W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GenASM-DC (64 PEs)</td>
<td>0.049</td>
<td>0.033</td>
</tr>
<tr>
<td>GenASM-TB</td>
<td>0.016</td>
<td>0.004</td>
</tr>
<tr>
<td>DC-SRAM (8 KB)</td>
<td>0.013</td>
<td>0.009</td>
</tr>
<tr>
<td>TB-SRAMs (64 x 1.5 KB)</td>
<td>0.055</td>
<td>0.049</td>
</tr>
</tbody>
</table>

Total (1 vault): 0.334 mm², 0.101 W
Total (32 vaults): 10.69 mm², 3.23 W
% of a Xeon CPU core: 1%
Key Results – Area and Power

- Based on our synthesis of GenASM-DC and GenASM-TB accelerator datapaths using the Synopsys Design Compiler with a 28nm LP process:
  - Both GenASM-DC and GenASM-TB operate @ 1GHz

**GenASM has low area and power overheads**
Use Cases of GenASM

1. **Reference genome**
   - **Indexing**
     - Hash table based index
   - **Seeding**
     - Candidate mapping locations
   - **Pre-Alignment Filtering**
     - Remaining candidate mapping locations
   - **Read Alignment**
     - Optimal alignment

2. **Reads from sequenced genome**
Use Cases of GenASM (cont’d.)

(1) Read Alignment Step of Read Mapping
   - Find the optimal alignment of how reads map to candidate reference regions

(2) Pre-Alignment Filtering for Short Reads
   - Quickly identify and filter out the unlikely candidate reference regions for each read

(3) Edit Distance Calculation
   - Measure the similarity or distance between two sequences

We also discuss other possible use cases of GenASM in our paper:
   - Read-to-read overlap finding, hash-table based indexing, whole genome alignment, generic text search
### Key Results

#### (1) Read Alignment
- **116x** speedup, **37x** less power than Minimap2 (state-of-the-art SW)
- **111x** speedup, **33x** less power than BWA-MEM (state-of-the-art SW)
- **3.9x** better throughput, **2.7x** less power than Darwin (state-of-the-art HW)
- **1.9x** better throughput, **82%** less logic power than GenAx (state-of-the-art HW)

#### (2) Pre-Alignment Filtering
- **3.7x** speedup, **1.7x** less power than Shouji (state-of-the-art HW)

#### (3) Edit Distance Calculation
- **22–12501x** speedup, **548–582x** less power than Edlib (state-of-the-art SW)
- **9.3–400x** speedup, **67x** less power than ASAP (state-of-the-art HW)
GenASM: A High-Performance, Low-Power Approximate String Matching Acceleration Framework for Genome Sequence Analysis

Damla Senol Cali
Carnegie Mellon University
(dsenol@andrew.cmu.edu)

Gurpreet S. Kalsi², Zulal Bingol³, Can Firtina⁴, Lavanya Subramanian⁵, Jeremie Kim¹,⁴, Rachata Ausavarungnirun⁶,¹, Mohammed Alser⁴, Juan Gomez-Luna⁴, Amirali Boroumand¹, Anant Nori², Allison Scibisz¹, Sreenivas Subramoney², Can Alkan³, Saugata Ghose⁷,¹, and Onur Mutlu⁴,¹,³