Genome Read In-Memory (GRIM) Filter:
Fast Location Filtering in DNA Read Mapping using Emerging Memory Technologies

Jeremie Kim[1], Damla Senol[1], Hongyi Xin[1], Donghyuk Lee[1], Mohammed Alser[2], Hasan Hassan[3], Oguz Ergin[3], Can Alkan[2] and Onur Mutlu[1]

[1] Carnegie Mellon University, [2] Bilkent University, [3] TOBB University of Economics and Technology

Read Mapping

Read Mapping: Mapping billions of DNA fragments (reads) against a reference genome to identify genomic variants

- Approximate string matching
- Computationally expensive alignment using quadratic-time **dynamic programming** algorithm
- Bottlenecked by memory bandwidth

Three types of read mappers:
- Suffix-array based mappers
- Hash table based mappers
- Hybrid

Hash Table Based Mappers

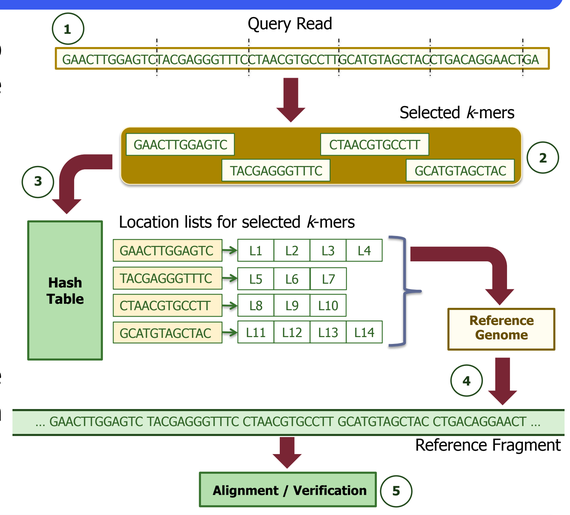
Seed-and-extend procedure to map reads against a reference genome allowing *e* indels.

- High **sensitivity**
- High **comprehensiveness**

BUT

- High **runtime**

The most recent fastest hash table based read mapper, ***mrFAST*** **with** ***FastHASH*** [Xin+, BMC Genomics 2013]



Problem

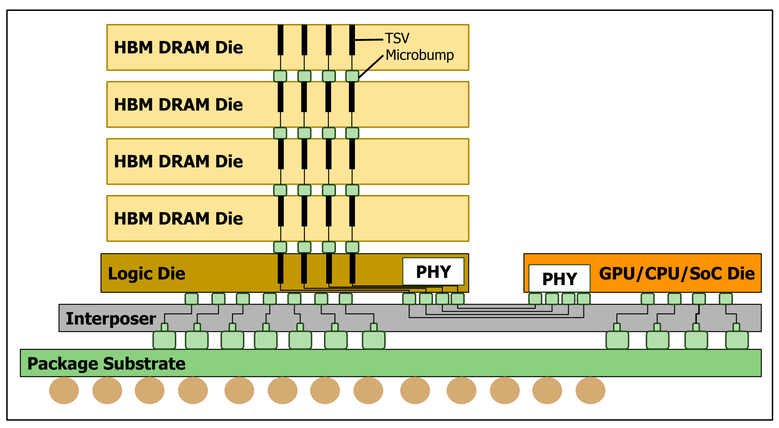
For lower runtimes, **location filters** can efficiently determine whether a candidate mapping location will result in an **incorrect mapping** ***before*** performing the computationally expensive **incorrect verification** by alignment. They should be **fast**.

Our Goal

Design and implement a new filter that **rejects incorrect mappings** before the alignment step
- Minimize the occurrences of unnecessary alignment
- Maintain **high sensitivity and comprehensiveness**
- Obtain low runtime and low false positive rate

Accelerate read mapping by overcoming the memory bottleneck by **utilizing 3D-stacked memory and its PIM capability** to handle data-intensive computation
- very fast and massively parallel operations on very large amounts of data **nearby memory**

3D-Stacked Logic-in-Memory DRAM



Recent technology that tightly couples memory and logic vertically with very high bandwidth connectors.

Small and numerous Through Silicon Vias (TSVs) connecting each layer, enable **higher bandwidth**, **lower latency** and **lower energy consumption**.

Logic layer enables **fast, massively parallel operations** on large sets of data, and provides the ability to run these operations **near memory** to alleviate the memory bottleneck.

Logic layer can be customized for application-specific accelerators.

GRIM-Filter Mechanism

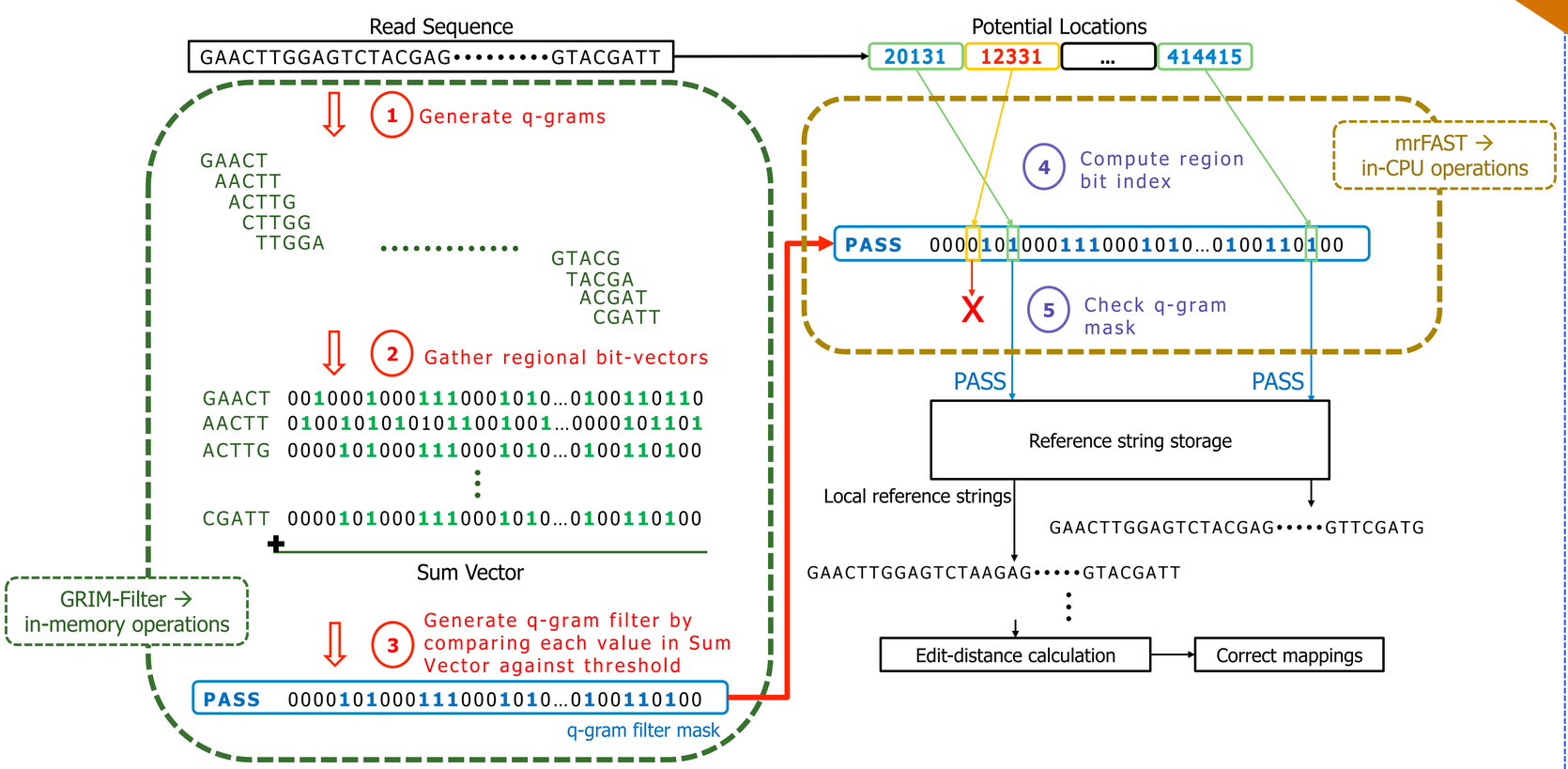
GRIM-Filter is based on two key ideas:
- Modify q-gram string matching to enable parallel checking for multiple locations, and
- Utilize a 3D-stacked DRAM architecture that both alleviates the memory bandwidth issue of our algorithm and parallelizes most of the filter.

GRIM-Filter has two main parts:

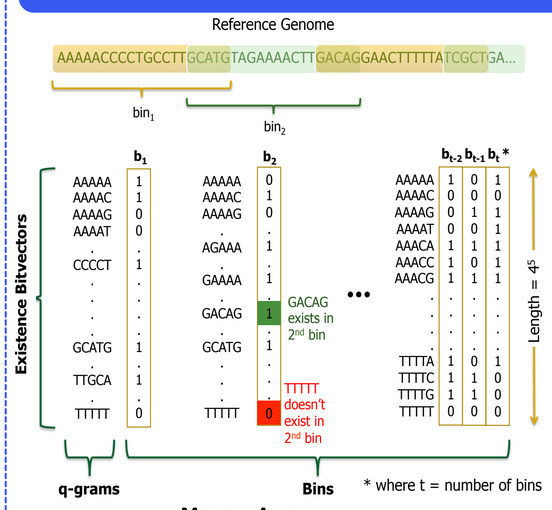
1) Precomputation: Divide the reference genome into consecutive bins and generate existence bitvectors for each bin.

2) Filtering Algorithm: Filter locations by quickly determining whether it is possible for a read to map to a specific segment of the genome.

Part 2: GRIM-Filter Walkthrough

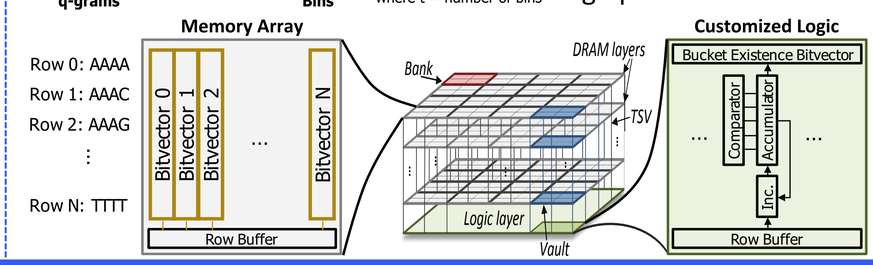


Part 1: Bins & Bitvectors

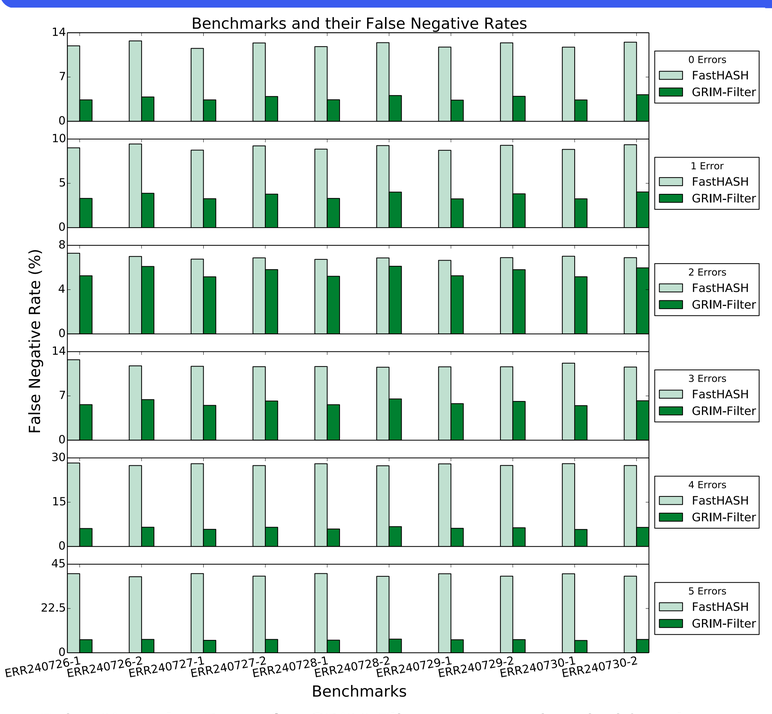


Existence bitvectors represent the existence of all possible permutations of **q-length** nucleotide sequences in each **bin**.

Bitvectors are distributed throughout the **memory layers** of 3D-stacked DRAM on top of the **customized logic layer**, which enables processing-in-memory and high parallelism.

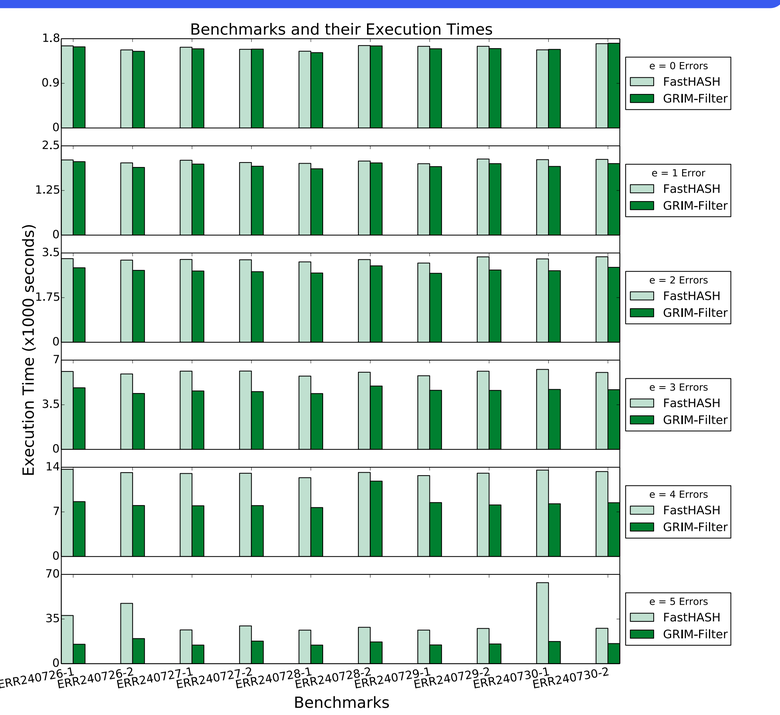


Results & Conclusion



False Negative Rates for GRIM-Filter as error threshold varies.

- Baseline**: mrFAST with FastHASH** mapper code [Xin+, BMC Genomics 2013]. However, GRIM-Filter **is fully complementary** to other mappers, too.
- **Key Results of GRIM-Filter:**
  - **5.59x-6.41x less false negative locations**, and
  - **1.81x-3.65x end-to-end speedup** over the state-of-the-art read mapper **mrFAST with FastHASH.**
- We show the inherent parallelism of our filter and ease of implementation for **3D-stacked memory**. There is great promise in adapting DNA read mapping algorithms to state-of-the-art and emerging memory and processing technologies.
- **Other Results:**
  - Examined sensitivity to **number of bins**: **450x65536**
  - Examined sensitivity to **q-gram size**: **5**
  - Found the best tradeoff between **memory consumption, filtering efficiency,** and **runtime**.



Runtimes for GRIM-Filter across the benchmarks as error threshold varies.