GRIM-Filter: Fast Seed Location Filtering in DNA Read Mapping using Emerging Memory Technologies

Jeremie Kim[1], Damla Senol[1], Hongyi Xin[1], Donghyuk Lee[1], Saugata Ghose[1], Mohammed Alser[2], Hasan Hassan[4,3], Oguz Ergin[3], Can Alkan[2] and Onur Mutlu[4,1]

[1] Carnegie Mellon [2] Bilkent University [3] TOBB University of Economics & Technology [4] ETH zürich

1: Read Mapping

Read Mapping: Mapping billions of DNA fragments (reads) against a reference genome to identify genomic variants
- Requires approximate string matching
- Computationally expensive alignment using **quadratic-time** dynamic programming algorithm
- **Bottlenecked by memory bandwidth**

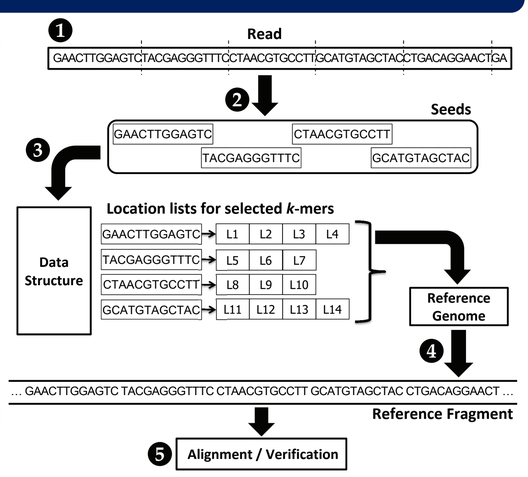
Three general types of read mappers:
- Suffix-array based mappers
- Hash table based mappers
- Hybrid

2: Hash Table Based Mappers

Seed-and-extend procedure to map reads against a reference genome allowing *e* indels. They have:
- High **sensitivity** (can tolerate many errors)
- High **comprehensiveness** (can find more mappings)

  BUT
- Low **speed**

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



3: Problem

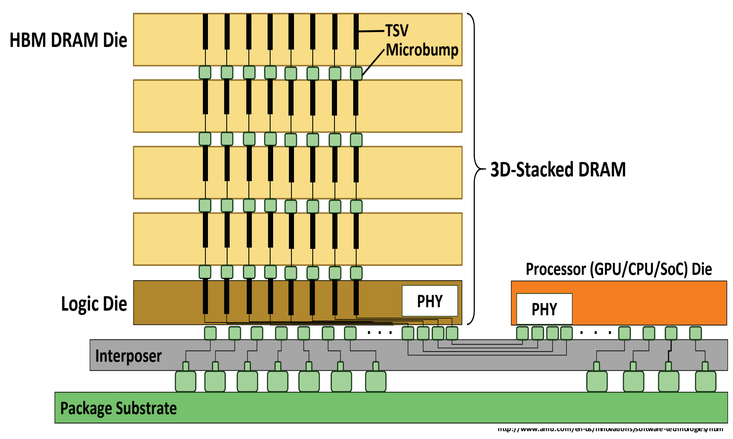
For lower runtimes, **location filters** can efficiently determine whether a candidate mapping location will result in an **incorrect mapping *before*** performing the computationally **expensive incorrect verification** by alignment. They should be **fast**.

4: 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 memory bottleneck with **3D-stacked memory and its PIM** for data-intensive computation
- Very fast parallel operations on big data sets **near memory**

5: 3D-Stacked Logic-in-Memory DRAM



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

Numerous Through Silicon Vias (TSVs) connecting layers, enable **higher bandwidth** and **lower latency** and **energy consumption**.

Customizable logic layer for application-specific accelerators.

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.

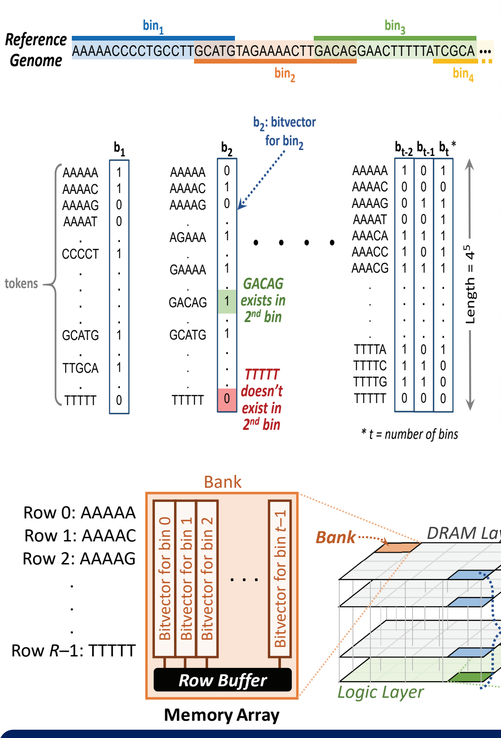
Processing in 3D-stacked memory is extremely good at accelerating embarrassingly parallel simple bit operations.

6: GRIM-Filter Mechanism

GRIM-Filter is based on two key ideas:
- Introduce **parallelism** to **q-gram string matching**
- Utilize a 3D-stacked DRAM to **alleviates** the memory bandwidth issue of our algorithm and **parallelizes** most of the filter.

GRIM-Filter has two main steps:
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 a read can map to a specific segment of the genome.

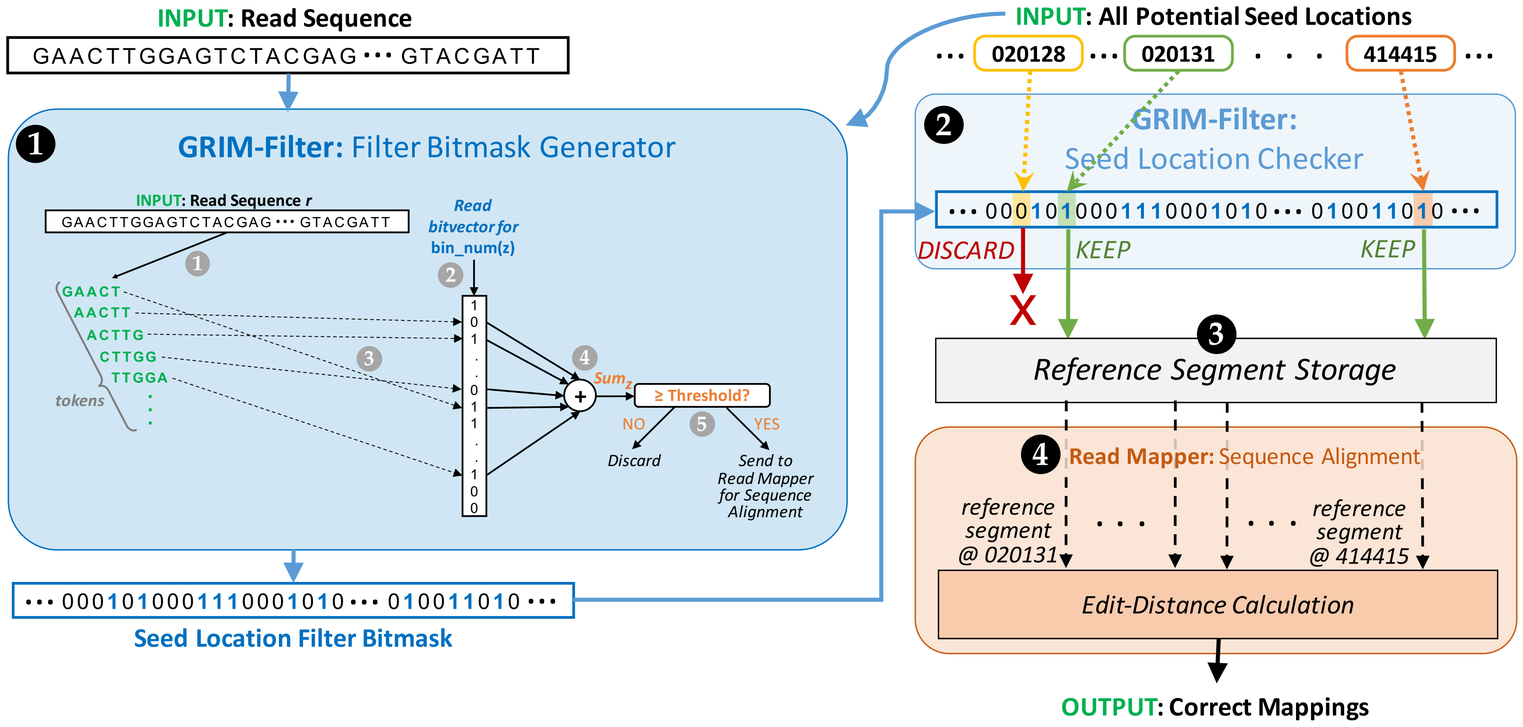
7: Bins & Bitvectors



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**.

8: GRIM-Filter Walkthrough



9: Results & Conclusion

- 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 to be the best tradeoff between **memory consumption, filtering efficiency,** and **runtime**.

*In the figures shown, smaller bars indicate better performance