# Previous Lectures

## Spring 2022 Meetings/Schedule

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Livestream</th>
<th>Meeting</th>
</tr>
</thead>
<tbody>
<tr>
<td>W1</td>
<td>8.3 Tue.</td>
<td>YouTube</td>
<td>M1: P&amp;S Mobile Genomics Course Introduction &amp; Project Proposals</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Live</td>
<td>(PDF) (PPT)</td>
</tr>
<tr>
<td>W2</td>
<td>15.3 Tue.</td>
<td>YouTube</td>
<td>M2: Introduction to Sequencing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Live</td>
<td>(PDF) (PPT)</td>
</tr>
<tr>
<td>W3</td>
<td>22.3 Tue.</td>
<td>YouTube</td>
<td>M3: Read Mapping</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Live</td>
<td>(PDF) (PPT)</td>
</tr>
</tbody>
</table>

[https://safari.ethz.ch/projects_and_seminars/spring2022/doku.php?id=genome_seq_mobile](https://safari.ethz.ch/projects_and_seminars/spring2022/doku.php?id=genome_seq_mobile)
Mohammed Alser, Hasan Hassan, Hongyi Xin, Oguz Ergin, Onur Mutlu, and Can Alkan
"GateKeeper: A New Hardware Architecture for Accelerating Pre-Alignment in DNA Short Read Mapping"
*Bioinformatics*, [published online, May 31], 2017.
[Source Code]
[Online link at Bioinformatics Journal]
GateKeeper: A New Hardware Architecture for Accelerating Pre-Alignment in DNA Short Read Mapping

Mohammed Alser, Hasan Hassan, Hongyi Xin, Oğuz Ergin, Onur Mutlu, Can Alkan

Bioinformatics, 2017

Presented by: Mohammed Alser
Executive Summary

- **Problem**: Genomic similarity measurement is a computational bottleneck. Examining the similarity of **highly-dissimilar genomic** sequences consumes an overwhelming majority of a modern read mapper’s execution time.

- **Goal**: Develop a fast and effective *filter* that can detect highly-dissimilar genomic sequences and eliminate them *before* invoking computationally costly alignment algorithms.

- **Key observation**: If two strings differ by $E$ edits, then every pairwise match can be aligned in at most $2E$ shifts.

- **Key ideas**:
  - Quickly find similar sequences using *Hamming Distance*.
  - Compute “*Shifted Hamming Distance*” for the rest of sequence pairs: ANDing $2E+1$ Hamming vectors of two strings, to identify dissimilar sequences.
  - Use only bit-parallel operations that nicely map to:
    - SIMD instructions, FPGA, Logic layer of the 3D-stacked memory, and In-memory accelerators (e.g., Ambit)

- **Key results**:
  - Provides a huge speedup of up to **130x** compared to the previous state of the art software solution.
We need intelligent algorithms and intelligent architectures that handle data well
Detailed Analysis of Tackling the Bottleneck

Mohammed Alser, Zülal Bingöl, Damla Senol Cali, Jeremie Kim, Saugata Ghose, Can Alkan, Onur Mutlu

“Accelerating Genome Analysis: A Primer on an Ongoing Journey”
Goal: Minimizing Alignment Time

Sequence Alignment is expensive

Our goal is to accelerate read mapping by reducing the need for dynamic programming algorithms
Key Idea

Genomic Strings

Dissimilar Strings

Ignore them if the number of differences exceeds a threshold.

Similar Strings

Find number, location, and type of differences?

EXPENSIVE!
1. Filter out most of dissimilar sequences.
2. Preserve all similar sequences.
3. Do it quickly.
Proposed Solution: GateKeeper

Pre-Alignment Filter + FPGA-based Alignment Filter = 1

$\times 10^{12}$ mappings

Query the Index

Low Speed & High Accuracy
Medium Speed, Medium Accuracy
High Speed, Low Accuracy

$\times 10^3$ mappings

1. High throughput DNA sequencing (HTS) technologies
2. Read Pre-Alignment Filtering: Fast & Low False Positive Rate
3. Read Alignment: Slow & Zero False Positives

Billions of Short Reads
Key observation:

- If two strings differ by $E$ edits, then every pairwise match can be aligned in at most $2E$ shifts.

Key ideas:

- *Quickly* find similar sequences using *Hamming Distance*.

- Compute "*Shifted Hamming Distance*": AND of $2E+1$ Hamming vectors of two strings, to identify invalid mappings

- Use only *bit-parallel operations* that nicely map to:
  - SIMD instructions
  - FPGA
  - Logic layer of the 3D-stacked memory
  - In-memory accelerators (e.g., Ambit)
Key observation:

- If two strings differ by $E$ edits, then every pairwise match can be aligned in at most $2E$ shifts.
Hamming Distance ($\sum \Theta$)

3 matches  5 mismatches

*Edit = 1 Deletion*

To cancel the effect of a deletion, we need to shift in the *right* direction.
Shifted Hamming Distance (Xin+ 2015)

ISTANBUL

0 0 0 1 1 1 1

1 1 1 0 0 0 0

0 0 0 1 0 0 0 0

7 matches 1 mismatches

Edit = 1 Deletion
Effect of Errors on Sequence Alignment

**Exact Matching:**

- TCCATTGACATTCCGTGAGCTGCTCCTTCTCTCCCCACCCCTTTGCCC
- TCCATTGACATTCCGTGAGCTGCTCCTTCTCTCCCCACCCCTTTGCCC

**Substitution:**

- TCCATTGACACTCGTGAGCTGCACTTTCTCTCCCCACCCCTTTGCCC
- TCCATTGACATTCCGTGAGCTGCTCCTTCTCTCCCCACCCCTTTGCCC

**Insertion:**

- TCCATTGACACCTCGTGAGCTGCTCCTTCTCTCCCACCCCTTTGCT
- TCCATTGACATTCCGTGAGCTGCTCCTTCTCTCCCCACCCCTTTGCCC

**Deletion:**

- TCCATTGACATTCCGAGCTGCTCCTTCTCTCCCACCCCTTTGCCC
- TCCATTGACATTCCGTGAGCTGCTCCTTCTCTCCCCACCCCTTTGCCC
Mechanisms

- **Key observation:**
  - If two strings differ by \( E \) edits, then every pairwise match can be aligned in at most \( 2E \) shifts.

- **Key ideas:**
  - *Quickly* find similar sequences using *Hamming Distance*.
  - Compute “Shifted Hamming Distance”: \( \text{AND of } 2E+1 \) Hamming vectors of two strings, to identify invalid mappings.
Our goal to track the diagonally consecutive matches in the neighborhood map.
GateKeeper

- **Key observation:**
  - If two strings differ by $E$ edits, then every pairwise match can be aligned in at most $2E$ shifts.

- **Key ideas:**
  - Quickly find similar sequences using *Hamming Distance*.
  - Compute "*Shifted Hamming Distance*": AND of $2E+1$ Hamming vectors of two strings, to identify invalid mappings.
  - Use only *bit-parallel operations* that nicely map to:
    - SIMD instructions
    - FPGA
    - Logic layer of the 3D-stacked memory
    - In-memory accelerators (e.g., Ambit)
Alignment Matrix vs. Neighborhood Map

**Needleman-Wunsch**

```
<table>
<thead>
<tr>
<th></th>
<th>C</th>
<th>T</th>
<th>A</th>
<th>T</th>
<th>A</th>
<th>T</th>
<th>A</th>
<th>C</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-1</td>
<td>-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-2</td>
<td>-2</td>
<td>-1</td>
<td>-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-2</td>
<td>-3</td>
<td>-2</td>
<td>-1</td>
<td>-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3</td>
<td>-3</td>
<td>-2</td>
<td>-1</td>
<td>-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-4</td>
<td>-3</td>
<td>-2</td>
<td>-1</td>
<td>-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-4</td>
<td>-3</td>
<td>-2</td>
<td>-2</td>
<td>-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

**Neighborhood Map**

```
<table>
<thead>
<tr>
<th></th>
<th>C</th>
<th>T</th>
<th>A</th>
<th>T</th>
<th>A</th>
<th>T</th>
<th>A</th>
<th>C</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Independent vectors can be processed in parallel using hardware technologies.
Hardware Architecture
GateKeeper Walkthrough (cont’d)

- E right-shift registers (length=ReadLength)
- E left-shift registers (length=ReadLength)
- \((2E+1) \times \text{ReadLength}\) 2-XOR operations.

Amend random zeros: \(101 \rightarrow 111 \) & \(1001 \rightarrow 1111\)

AND all masks, ACCEPT iff number of ‘1’ \(\leq\) Threshold

- \((2E) \times \text{ReadLength}\) 2-AND operations.
- \((\text{ReadLength}/4)\) 5-input LUT.
- \(\log_2 \text{ReadLength}\)-bit counter.

Hamming mask

\[
\begin{array}{cccccccc}
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\
\end{array}
\]

Hamming mask after amending

\[
\begin{array}{cccccccc}
0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\
1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
\end{array}
\]

- \((2E+1) \times \text{ReadLength}\) 5-input LUT.
Configurable logic blocks (CLBs) are the main logic resources for implementing sequential as well as combinatorial circuits.
Virtex-7 FPGA Layout

The LUTs in 7 series FPGAs can be configured as either a 6-input LUT with one output, or as two 5-input LUTs with separate outputs.

*Figure 1-1: Arrangement of Slices within the CLB*

*Table 2-1: Logic Resources in One CLB*

<table>
<thead>
<tr>
<th>Slices</th>
<th>LUTs</th>
<th>Flip-Flops</th>
<th>Arithmetic and Carry Chains</th>
<th>Distributed RAM&lt;sup&gt;(1)&lt;/sup&gt;</th>
<th>Shift Registers&lt;sup&gt;(1)&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>8</td>
<td>16</td>
<td>2</td>
<td>256 bits</td>
<td>128 bits</td>
</tr>
</tbody>
</table>

Key Results:
Methodology and Evaluation
Methodology

- System setup:
  - 3.6 GHz Intel i7-3820 (supports only PCIe 2.0)
  - Xilinx VC709 (~$5000)
    - Architecture implementation using Vivado 2014.4 in Verilog
    - RIFFA 2.2 to perform Host-FPGA PCIe communication

- Evaluated dataset:
  - Real sequencing read set (ERR240727_1.fastq)
  - Five simulated read sets of 100 bp and 300 bp long Illumina-like reads with different type and number of edits.
Prior Work on Pre-Alignment Filtering

- **Adjacency Filter** *(BMC Genomics, 2013)*
  - Slow
  - Accepts a large number of dissimilar sequences.

- **Shifted Hamming Distance (SHD)** *(Bioinformatics, 2015)*
  - It requires the same execution time as the Adjacency Filter
  - It accepts 4X fewer dissimilar sequences compared to the Adjacency Filter.
  - It suffers from a limited sequence length (≤ 128 bp)
VC709 Resource Utilization

Theoretically:

- Up to **140 GateKeeper Processing cores** on a single FPGA (E=5, 100bp)
- **BUT bottlenecked** by PCIe bandwidth
- **Small area allows integration** into FPGAs already inside of sequencers

<table>
<thead>
<tr>
<th>Table 2. FPGA resource utilization for a single GateKeeper core</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read length</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Edit distance</td>
</tr>
<tr>
<td>Slice LUT&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
<tr>
<td>Slice Register&lt;sup&gt;b&lt;/sup&gt;</td>
</tr>
</tbody>
</table>

<sup>a</sup>LUT: look-up tables.

<sup>b</sup>Flip-flop.
VC709 Resource Utilization

Experimentally:

- GateKeeper aligns each read against up to 8 and 16 different reference segments in parallel, without violating the timing constraints for a sequence lengths of 300 and 100 bp, respectively.

Table 3. Overall system resource utilization under different read lengths and edit distance thresholds

<table>
<thead>
<tr>
<th>Read length</th>
<th>100 bp</th>
<th>300 bp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit distance</td>
<td>16 GateKeeper cores</td>
<td>8 GateKeeper cores</td>
</tr>
<tr>
<td>2</td>
<td>32%</td>
<td>50%</td>
</tr>
<tr>
<td>5</td>
<td>45%</td>
<td>69%</td>
</tr>
<tr>
<td>Slice LUT</td>
<td>2</td>
<td>50%</td>
</tr>
<tr>
<td>Slice register</td>
<td>2</td>
<td>17%</td>
</tr>
<tr>
<td>Block memory</td>
<td>2</td>
<td>2%</td>
</tr>
</tbody>
</table>
**GateKeeper Accelerator Architecture**

- **Maximum data throughput** = \( \sim 13.3 \) billion bases/sec
- Can examine **8 (300 bp) or 16 (100 bp) mappings concurrently** at 250 MHz
- **Occupies 50\%** (100 bp) to **91\%** (300 bp) of the FPGA slice LUTs and registers
FPGA Chip Layout

GateKeeper: 17.6%, PCIe Controller, RIFFA, and IO: 5%

300 bp
E=15
Speed & Accuracy Results

90x-130x faster than SHD (Xin et al., 2015) and the Adjacency Filter (Xin et al., 2013).

Accepts 4x fewer dissimilar strings than the Adjacency Filter (Xin et al., 2013).

10x speedup with the addition of GateKeeper to the mrFAST mapper (Alkan et al., 2009).

Freely available online
github.com/BilkentCompGen/GateKeeper
GateKeeper Conclusions

- There is a significant performance gap between high-throughput DNA sequencers and read mapper

- Sequence alignment is *computationally expensive* and *unavoidable*

- **GateKeeper** is the *first hardware accelerator architecture* (as a pre-alignment filter) for *quickly* rejecting dissimilar sequences

- It provides a huge speedup of up to **130x** compared to the previous state of the art software solution.
GateKeeper Conclusions

- FPGA-based pre-alignment filtering greatly speeds up read mapping
  - 10x speedup of a state-of-the-art mapper (mrFAST)

- FPGA-based pre-alignment can be integrated with the sequencer
  - It can help to hide the complexity and details of the FPGA
  - Enables real-time filtering while sequencing
More on SHD (SIMD Implementation)

- Download and test for yourself
- https://github.com/CMU-SAFARI/Shifted-Hamming-Distance

Sequence analysis

**Shifted Hamming distance: a fast and accurate SIMD-friendly filter to accelerate alignment verification in read mapping**

Hongyi Xin\(^1\),\(^*\), John Greth\(^2\), John Emmons\(^2\), Gennady Pekhimenko\(^1\), Carl Kingsford\(^3\), Can Alkan\(^4\),\(^*\) and Onur Mutlu\(^2\),\(^*\)
More on GateKeeper

- Download and test for yourself
  https://github.com/BilkentCompGen/GateKeeper

Alser+, "GateKeeper: A New Hardware Architecture for Accelerating Pre-Alignment in DNA Short Read Mapping", Bioinformatics, 2017.
Strengths

- **New and simple** solution to a critical problem. New algorithm and hardware architecture.

- GateKeeper **does not sacrifice** any of the aligner capabilities, as it does not modify or replace the alignment step.

- Design is **scalable**; could add more processing cores in the future.

- Some sequencers use **FPGAs** as well, so GateKeeper could be integrated into them.
Strengths (cont’d)

- Authors understand and highlight limitations of GateKeeper
- Greatly improves filtering speed and accuracy
- Spurred quite a few papers that build on GateKeeper
- Well-written, interesting and easy to understand paper
Weaknesses

- The benefits of such a mechanism require an FPGA and advanced knowledge with computers, this may be problematic for some biologists/genomicists/geneticists.

- The amendment of the random zeros is a simple “hack” to reduce the number of false positives, but there is no explanation why GateKeeper only flips the patterns 101 and 1001, what about 10001? And $10^n1$?

- The paper can be confusing at times due to the use of a “supplementary material” document that is constantly referred to (but understandable as there was a page limit set by the publication journal).
Weaknesses (cont’d)

- GateKeeper’s accuracy degrades exponentially for $E > 2\%$, and becomes ineffective for $E > 8\%$.

- GateKeeper is tested using short reads
  - 3rd generation sequencing machines produce much longer reads
Thoughts and Ideas
Accelerating Read Mapping

Accelerating Indexing
- Reducing the number of seeds
- Reducing data movement during indexing

Accelerating Pre-Alignment Filtering
- q-gram filtering
- Pigeonhole principle
- Base counting
- Sparse DP

Accelerating Alignment
- Accurate alignment accelerators
- Heuristic-based alignment accelerators

Our Ongoing Journey

Near-memory/In-memory Pre-alignment Filtering

GRIM-Filter [BMC Genomics’18]
SneakySnake [IEEE Micro’21]
GenASM [MICRO 2020]

Near-memory Sequence Alignment

GenASM [MICRO 2020]

Specialized Pre-alignment Filtering Accelerators (GPU, FPGA)

GateKeeper [Bioinformatics’17]
MAGNET [AACBB’18]
Shouji [Bioinformatics’19]
GateKeeper-GPU [arXiv’21]
SneakySnake [Bioinformatics’20]
Key Takeaways

- A novel method to accelerate Sequence Alignment in genome analysis.
- Simple and effective
- Hardware/software cooperative
- Good potential for work building on it to extend it
  - To make things more efficient and effective
  - Multiple works have already built on the paper (see MAGNET, Shouji, GRIM-Filter, SneakySnake, GenCache)
- Easy to read and understand paper
Adoption of hardware accelerators in genome analysis
I have a major concern with the work that is actually not a problem with the manuscript at all. Specifically, I have the concern that there has been little to no adoption of previous specialized hardware solutions related to improving the speed of alignment. While there has been considerable work in this area (which the authors do an admirable job of citing), it does not seem that these hardware-based solutions have gained any type of real traction in the community, as the vast majority of alignment is still performed on “regular” CPUs, where the extent of hardware acceleration is the adoption of specific SIMD or vectorized instructions. While I don’t think that this practical concern should preclude publication of the current work, it is something worth considering (e.g. what, if any, of the proposed improvements to the SHD filter could be “back-ported” to a software-only solution).
Our Response

We see the reviewer’s point, but we do not believe this should be held against the research in the area of FPGA-based acceleration of read mapping in particular or genomics in general. It always takes time to adopt a “new” or “different” hardware technology since it requires investment into the hardware infrastructure. The main challenges/barriers that limit the popularity of FPGAs in the genomics field are the high cost, design effort, and development time. Due to the fact that the deliverable of such projects is normally a hardware product, researchers tend to commercialize their research with startup companies and engage themselves with industrial collaborators, as we describe below. Today, the cost structure of FPGAs is changing because major cloud infrastructures (e.g., by Microsoft Azure and Amazon AWS) offer FPGAs as core engines of the infrastructure. Therefore, we believe the benefits of FPGA-based acceleration has become available to many more folks in the community, especially with the open-source release of such FPGA-accelerated solutions. To increase adoption, we have decided to release our source code for GateKeeper. It is available on https://github.com/BilkentCompGen/GateKeeper.

Some examples of the research groups that commercialize their research and promote FPGA-based or even cloud-based products for genomics are as follows:
http://www.timelogic.com/catalog/775
http://www.edicogenome.com/dragen_bioit_platform/the-dragen-engine-2/
http://www.bcgsc.ca/platform/bioinfo/software/XpressAlign/releases/1.0
https://www.sevenbridges.com/amazon/
Our Response (cont’d)

It is also important to emphasize that the necessity of designing a mapper on hardware is currently steering the field towards more personalized medicine. Hardware-accelerated mappers (using various platforms such as SIMD, GPUs, and FPGAs) are becoming increasingly popular as they can be potentially directly integrated into sequencing machines (the Illumina sequencer, for example, includes an FPGA chip inside it https://support.illumina.com/content/dam illumina-support/documents/downloads/software/hiseq/hcs_2-0-12/installnotes_hcs2-0-12.pdf ), such that we have a single machine that can perform both sequencing and mapping (Lindner, et al., Bioinformatics 2016). This approach has two benefits. First, it can hide the complexity and details of the underlying hardware from users who are not necessarily aware about FPGAs (e.g., biologists and mathematicians). Second, it allows a significant reduction in total genome analysis time by starting read mapping while still sequencing. Hence, an end user or researcher in genomics might not directly deal with the “pre-alignment on FPGA” or “mapper on FPGA”, but they might purchase a sequencer that performs pre-alignment and alignment using FPGAs inside. As such, one potential target of our research is to influence the design of more intelligent sequencing machines by integrating GateKeeper inside them.

In fact, we believe GateKeeper is very suitable to be used as part of a sequencer as it provides a complete pre-alignment system that includes many processing cores, where all processing cores work in parallel to provide extremely fast filtering. We believe such a fast approach can make sequencers more intelligent and attractive.
Remember What We Said in the First Lecture

Dream

and, they will come

• Computing landscape is very different from 10-20 years ago.
• As applications push boundaries, computing platforms will become increasingly strained.
Illumina DRAGEN Bio-IT Platform (2018)

- Processes whole genome at 30x coverage in ~25 minutes with hardware support for data compression

emea.illumina.com/products/by-type/informatics-products/dragen-bio-it-platform.html
NVIDIA Clara Parabricks (2020)

A University of Michigan’s startup in 2018 and joined NVIDIA in 2020

https://developer.nvidia.com/clara-parabricks
Computing is Still Bottlenecked by Data Movement
Processing Genomic Data Where it Makes Sense

Intelligent Genome Analysis

FPGAs

Heterogeneous Processors and Accelerators

(General Purpose) GPUs

Hybrid Main Memory

Sequencing Machine

Persistent Memory/Storage
Most speedup comes from parallelism enabled by novel architectures and algorithms
More on GateKeeper [Alser+, Bioinformatics 2017]

Mohammed Alser, Hasan Hassan, Hongyi Xin, Oguz Ergin, Onur Mutlu, and Can Alkan
"GateKeeper: A New Hardware Architecture for Accelerating Pre-Alignment in DNA Short Read Mapping"
Bioinformatics, [published online, May 31], 2017.
[Source Code]
[Online link at Bioinformatics Journal]
Mohammed Alser, Jeremy Rotman, Dhrithi Deshpande, Kodi Taraszka, Huwenbo Shi, Pelin Icer Baykal, Harry Taegyun Yang, Victor Xue, Sergey Knyazev, Benjamin D. Singer, Brunilda Balliu, David Koslicki, Pavel Skums, Alex Zelikovsky, Can Alkan, Onur Mutlu, Serghei Mangul

"Technology dictates algorithms: Recent developments in read alignment"

Genome Biology, 2021

[Source code]
Accelerating Read Mapping

Detailed Analysis of Tackling the Bottleneck

Mohammed Alser, Zülal Bingöl, Damla Senol Cali, Jeremie Kim, Saugata Ghose, Can Alkan, Onur Mutlu

“Accelerating Genome Analysis: A Primer on an Ongoing Journey”
More on Fast Genome Analysis …

- Onur Mutlu,
  "Accelerating Genome Analysis: A Primer on an Ongoing Journey"
  Invited Lecture at Technion, Virtual, 26 January 2021.
  [Slides (pptx) (pdf)]
  [Talk Video (1 hour 37 minutes, including Q&A)]
  [Related Invited Paper (at IEEE Micro, 2020)]
More on Intelligent Genome Analysis …

Our Solution: GateKeeper

Alignment Filter + FPGA-based Alignment Filter = 1st

x10^{12} mappings

x10^3 mappings

Low Speed & High Accuracy
Medium Speed, Medium Accuracy
High Speed, Low Accuracy

Read Pre-Alignment Filtering
Fast & Low False Positive Rate
Read Alignment
Slow & Zero False Positives

https://www.youtube.com/watch?v=ygmQpdDTL7o
Detailed Lectures on Genome Analysis

- Computer Architecture, Fall 2020, Lecture 3a
  - **Introduction to Genome Sequence Analysis** (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=CrRb32v7SJc&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=5](https://www.youtube.com/watch?v=CrRb32v7SJc&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=5)

- Computer Architecture, Fall 2020, Lecture 8
  - **Intelligent Genome Analysis** (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=ygmQpdDTL7o&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=14](https://www.youtube.com/watch?v=ygmQpdDTL7o&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=14)

- Computer Architecture, Fall 2020, Lecture 9a
  - **GenASM: Approx. String Matching Accelerator** (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=XoLpzmN-Pas&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=15](https://www.youtube.com/watch?v=XoLpzmN-Pas&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=15)

- Accelerating Genomics Project Course, Fall 2020, Lecture 1
  - **Accelerating Genomics** (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=rgjl8ZyLsAg&list=PL5Q2soXY2Zi9E2bBVAgCqLgwiDRQDTyId](https://www.youtube.com/watch?v=rgjl8ZyLsAg&list=PL5Q2soXY2Zi9E2bBVAgCqLgwiDRQDTyId)

[https://www.youtube.com/onurmutlulectures](https://www.youtube.com/onurmutlulectures)
Prior Research on Genome Analysis (1/2)

- Alser + "SneakySnake: A Fast and Accurate Universal Genome Pre-Alignment Filter for CPUs, GPUs, and FPGAs.", Bioinformatics, 2020.


Prior Research on Genome Analysis (2/2)


- Alser+, "GateKeeper: A New Hardware Architecture for Accelerating Pre-Alignment in DNA Short Read Mapping”, Bioinformatics, 2017.

GateKeeper: A New Hardware Architecture for Accelerating Pre-Alignment in DNA Short Read Mapping

Mohammed Alser, Hasan Hassan, Hongyi Xin, Oğuz Ergin, Onur Mutlu, Can Alkan
Bioinformatics, 2017

Presented by: Mohammed Alser