Seminar in Computer Architecture
Intelligent Genomic Analyses

Dr. Mohammed Alser
@mealser

ETH Zurich
Spring 2023
2 March 2023
Intelligent Genome Analysis

**Mohammed Alser**, Joel Lindegger, Can Firtina, Nour Almadhoun, Haiyu Mao, Gagandeep Singh, Juan Gomez-Luna, Onur Mutlu

"From Molecules to Genomic Variations: Intelligent Algorithms and Architectures for Intelligent Genome Analysis"

Computational and Structural Biotechnology Journal, 2022

[Source code]

**Review**

*From molecules to genomic variations: Accelerating genome analysis via intelligent algorithms and architectures*

Mohammed Alser *, Joel Lindegger, Can Firtina, Nour Almadhoun, Haiyu Mao, Gagandeep Singh, Juan Gomez-Luna, Onur Mutlu *

*ETH Zurich, Gloriastrasse 35, 8092 Zürich, Switzerland*
What is a Genome?

An organism’s complete set of genetic instructions
How Large is a Genome?

Prime Tower, Zurich

~3.2 billion genomic bases
How About Other Species?

- Phi X174 virus: 5.386 Kilo bp
- E. coli O157:H7: 5.44 Million bp
- Homo Sapiens: 3.2 Billion bp
- Onion, Allium Cepa: 16 Billion bp
- Paris Japonica: 149 Billion bp
How to Analyze a Genome?

**NO**

machine gives the **complete** sequence of genome as output
DNA Testing

Health + Ancestry Service

$199

- Includes everything in Ancestry + Traits Service

PLUS

- 10+ Health Predisposition reports*
- 5+ Wellness reports
- 40+ Carrier Status reports*

SAFARI

https://www.myheritage.ch/dna  https://www.23andme.com/
Sequencing Technologies

Oxford Nanopore (ONT)

PacBio (HiFi, CLR)

Illumina

... and more! All produce data with different properties.
Genome Sequencer is a Chopper

Regardless the sequencing machine, reads still lack information about their order and location (which part of genome they are originated from)
Solving the Puzzle

Reference genome

Reads

Types of Genomic Reads

Wenger+, "Accurate circular consensus long-read sequencing improves variant detection and assembly of a human genome", *Nature Biotechnology*, 2019

https://labs.wsu.edu/genomicscore/illumina-sequencing/
https://pacbio.gs.washington.edu/
Genome Analysis in Real Life

Sample Collection

DNA Molecule

Chopped DNA Fragments

Sequencing

Raw Sequencing Data

Computational Steps

Library Preparation

Genomic Analyses
Different Raw Sequencing Data

Illumina

Multiple images

.BCL/.CBCL

ONT

Squiggle

.FAST5

PacBio

30-hour movie

.BAM
Sequencing in Action

![MinION](https://store.nanoporetech.com/flow-cell-r9-4-1.html)

**Chemistry type:**

<table>
<thead>
<tr>
<th>Pack size</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Flow cell</td>
<td>$900.00</td>
</tr>
<tr>
<td>12 Flow cells</td>
<td>$9,480.00</td>
</tr>
</tbody>
</table>

Portable DNA/RNA sequencing for anyone
How Does Nanopore Machine Work?

- **Nanopore** is a nano-scale hole (<20nm).

https://www.yourgenome.org/facts/what-is-oxford-nanopore-technology-ont-sequencing/
How Does Nanopore Machine Work?

Nanopore is a nano-scale hole (<20nm).

Check Nanopore virtual tour:
https://nanoporetech.com/resource-centre/minion-video

Graphene nanopore

DNA
Changes in sequencing technologies can render some read mapping algorithms irrelevant.
Read Mapping in 111 pages!

In-depth analysis of 107 read mappers (1988-2020)

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]
Solving the Puzzle

Reference genome

Reads

Obtaining the Human Reference Genome

- **GRCh38.p13**
- Description: Genome Reference Consortium Human Build 38 patch release 13 (GRCh38.p13)
- Organism name: *Homo sapiens (human)*
- Date: 2019/02/28
- 3,099,706,404 bases
- Compressed .fna file (964.9 MB)
Obtaining .FASTQ Files


ERX215261: Whole Genome Sequencing of human TSI NA20754
1 ILLUMINA (Illumina HiSeq 2000) run: 4.1M spots, 818.7M bases, 387.2Mb downloads

Design: Illumina sequencing of library 6511095, constructed from sample accession SRS001721 for study accession SRP000540. This is part of an Illumina multiplexed sequencing run (9340_1). This submission includes reads tagged with the sequence TTAGGCAT.

Submitted by: The Wellcome Trust Sanger Institute (SC)

Study: Whole genome sequencing of (TSI) Toscani in Italia HapMap population
  PRJNA333847 • SRP000540 • All experiments • All runs

Sample: Coriell GM20754
  SAMN00001273 • SRS001721 • All experiments • All runs
  Organism: Homo sapiens

Library:
  Name: 6511095
  Instrument: Illumina HiSeq 2000
  Strategy: WGS
  Source: GENOMIC
  Selection: RANDOM
  Layout: PAIRED
  Construction protocol: Standard

Runs: 1 run, 4.1M spots, 818.7M bases, 387.2Mb
Matching Each Read with Reference Genome

.FASTA file:

>NG_008679.1:5001-38170 Homo sapiens paired box 6 (PAX6)
ACCCTTCATTTGACATTTAATCTCTGGGCGAGGGAAGGCTGTCAGATCTCT
GCCACCTCCCCTGCCCCAGGCGGCCTGAGAAGGTGTGGGAACCGCCTGCTACCTGCTCCTCCTCCCGC
CCTCCGCTTCGTTCAACCGCCGCCCGGGCCGGCTGCTGGCCGGCGGGCGCTTGCAACCTGCTCCGCTG
CCAGCGACTGCTGTCCTAAAAATCAAAGCCGCCCCCAAGTGCGCCCCGGGGCTTGTATTTTGGCTTTAAG
GAGGCGATAACAGATGGGAAGCAGTTACTGAGGGAGGGATAGGAAAGGGGGTTGGAGGAGGACTTGTCTTT
TGCCGAGTAGTTTAAATGAAAGAACTCCTACTCTCCATCCGACGCTCGAGCTCAGTCTCCCGCCCT
GAGCTGGAGTAGGGGGCGGACTGCTGCTGCTGCTGCTGCTGCTAAGCCACTCGCGACCACCAGAAAATGCA
GGAGGTGGGACGCACTTTGGCATCCGGACCTCCTCTCTCTCTCAGCAGAGTTCACGCTTGGGAAG
TCCGTACCGCGCCTTTTTAAAGACACCCCTGGCCGCGGTTGCGGGCGGAGGTGCAGCAGAAGTTTCCCG
GCGTTGCAAGATGTCAGATGGGCTGGGTTCTCTTCTCAGAAAGACGC

.FASTQ file:

@HWI-EAS209_0006_FC706VJ:5:58:5894:21141#ATCACG/1
T[ATTAATCTTTAGATN]NNNNNNNNTAG
+HWI-EAS209_0006_FC706VJ:5:58:5894:21141#ATCACG/1
efffffffffcefeffccfffeffdfe`feed`}_Ba^__[YBBBBBBBBBBRTT
Genome Index Properties

- The index is built **only once** for each reference.

- **Seeds** can be overlapping, non-overlapping, spaced, adjacent, Syncmers, Strobemers, BLEND, non-adjacent, minimizers, compressed, ...

<table>
<thead>
<tr>
<th>Tool</th>
<th>Version</th>
<th>Index Size</th>
<th>Indexing Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>mrFAST</td>
<td>2.2.5</td>
<td>16.5 GB</td>
<td>20.00 min</td>
</tr>
<tr>
<td>minimap2</td>
<td>0.12.7</td>
<td>7.2 GB</td>
<td>3.33 min</td>
</tr>
<tr>
<td>BWA-MEM</td>
<td>0.7.17</td>
<td>4.7 GB</td>
<td>49.96 min</td>
</tr>
</tbody>
</table>

*Human genome = 3.2 GB*
Base-by-Base Comparison

read 1: $\text{CCTTAGTATATATACTAGTAC}$

ref 1: $\text{CGTAGCATAATCCTAGTAC}$

reference segment that spans locations (5, 7, and 9)
**Edit distance** is defined as the minimum number of edits (i.e. insertions, deletions, or substitutions) needed to make the read exactly matches the reference segment.

- **organization x operation**
  - Ref: organization
  - Read: operation
  - Edit distance = 7

- **organization x translation**
  - Ref: organization
  - Read: translation
  - Edit distance = 4
Analysis is Bottlenecked in Read Mapping!!

48 Human whole genomes at 30× coverage in about 2 days

32 CPU hours on a 48-core processor

Need for Speed
Finding SNPs Associated with Complex Trait

| Individual #1 | SNP1: ...ACATG**CGACATTTCATA**AGGCC... | SNP2: ...CGACATTTCATA**AGGCC... | Blood Pressure: 180 |
| Individual #2 | ...ACATG**CGACATTTCATA**AGGCC... | 175 |
| Individual #3 | ...ACATG**CGACATTTCATA**AGGCC... | 170 |
| Individual #4 | ...ACATG**CGACATTTCATA**AGGCC... | 165 |
| Individual #5 | ...ACATG**CGACATTTCATA**AGGCC... | 160 |
| Individual #6 | ...ACATG**CGACATTTCATA**AGGCC... | 145 |
| Individual #7 | ...ACATG**CGACATTTCATA**AGGCC... | 140 |
| Individual #8 | ...ACATG**CGACATTTCATA**AGGCC... | 130 |
| Individual #9 | ...ACATG**TCGACATTTCATA**AGGCC... | 120 |
| Individual #10 | ...ACATG**TCGACATTTCATA**AGGCC... | 120 |
| Individual #11 | ...ACATG**TCGACATTTCATA**AGGCC... | 115 |
| Individual #12 | ...ACATG**TCGACATTTCATA**AGGCC... | 110 |
| Individual #13 | ...ACATG**TCGACATTTCATA**AGGCC... | 110 |
| Individual #14 | ...ACATG**TCGACATTTCATA**AGGCC... | 110 |
| Individual #15 | ...ACATG**TCGACATTTCATA**AGGCC... | 105 |
| Individual #16 | ...ACATG**TCGACATTTCATA**AGGCC... | 100 |

SNP: single nucleotide polymorphism

**SAFARI** Discovering the Causal Variants Involved in GWAS Studies, CGSI 2018, UCLA
Much Larger Structural Variations!

**AUTISM**
Deletion of 593 kb

**OBESITY**
Walters, *Nature* 2010
Deletion of 593 kb

**SCHIZOPHRENIA**
McCarthy, *Nat Genet* 2009
Duplication of 593 kb

**UNDERWEIGHT**
Duplication of 593 kb

Deletion in the short arm of chromosome 16 (16p11.2)

Duplication in the short arm of chromosome 16 (16p11.2)
SNPs and Personalized Medicine

SNP rs12979860

Basic Information

<table>
<thead>
<tr>
<th>Name</th>
<th>rs12979860</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chromosome</td>
<td>19</td>
</tr>
<tr>
<td>Position</td>
<td>39248147</td>
</tr>
<tr>
<td>Weight of evidence</td>
<td>926</td>
</tr>
</tbody>
</table>

Allele Frequency

- A: 49%
- T: 27%
- G: 23%
- C: 4%
- -: 1%
- 0: 0%

Links to SNPedia

<table>
<thead>
<tr>
<th>Title</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>rs12979860 T/T</td>
<td>~20-25% of such hepatitis c patients respond to treatment</td>
</tr>
<tr>
<td>rs12979860 C/C</td>
<td>~80% of such hepatitis c patients respond to treatment</td>
</tr>
<tr>
<td>rs12979860 C/T</td>
<td>~20-40% of such hepatitis c patients respond to treatment</td>
</tr>
</tbody>
</table>

https://opensnp.org/snps/rs12979860
rWGS can be performed in 2-day (costly) or 5-day time to interpretation.

Diagnostic rWGS for infants
- Avoids morbidity
- Reduces hospital stay length by 6%-69%
- Reduces inpatient cost by $800,000-$2,000,000.
“From 2019, all seriously ill children in UK will be offered whole genome sequencing as part of their care”
Characterizing genomic variations of 49,962 Icelanders took **4.15 million CPU hours** or 83 CPU hours per sample on average.

“GraphTyper2 enables population-scale genotyping of structural variation using pangenome graphs”, Nature Communications, 2019
Goal: What organisms are present in a given environment and how abundant are they?
Petabase-scale Viral Discovery

Building and Profiling 3,500 genomic assemblies needs 28,000 virtual AWS CPUs.

Edgar+, "Petabase-scale sequence alignment catalyses viral discovery", Nature 2022

https://serratus.io/
Population-Scale Microbiome Profiling

Danko+, "A global metagenomic map of urban microbiomes and antimicrobial resistance", Cell, 2021
We Need Faster & Scalable Genome Analysis

Understanding **genetic variations**

Predicting the presence and relative abundances of **microbes** in a sample

Rapid surveillance of **disease outbreaks**

Developing **personalized medicine**

And many other applications …
What makes read mapping a bottleneck?
# A Tsunami of Sequencing Data

## A Tera-scale increase in sequencing production in the past 25 years

<table>
<thead>
<tr>
<th>Category</th>
<th>Year</th>
<th>Scale</th>
</tr>
</thead>
<tbody>
<tr>
<td>Genes &amp; Operons</td>
<td>1990</td>
<td>Kilo</td>
</tr>
<tr>
<td>Bacterial genomes</td>
<td>1995</td>
<td>Mega</td>
</tr>
<tr>
<td>Human genome</td>
<td>2000</td>
<td>Giga</td>
</tr>
<tr>
<td>Human microbiome</td>
<td>2005</td>
<td>Tera</td>
</tr>
<tr>
<td>50K Microbiomes</td>
<td>2015</td>
<td>Peta</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Category</th>
<th>Year</th>
<th>Scale</th>
</tr>
</thead>
<tbody>
<tr>
<td>200K Microbiomes</td>
<td>2020</td>
<td>Exa</td>
</tr>
<tr>
<td>1M Microbiomes</td>
<td>2025</td>
<td>Zetta</td>
</tr>
<tr>
<td>Earth Microbiome</td>
<td>2030</td>
<td>Yotta</td>
</tr>
</tbody>
</table>

**what is expected for the next 15 years? (a Giga?)**

- Exa = 1,000,000,000,000,000,000,000
- Zetta = 1,000,000,000,000,000,000,000
- Yotta = 1,000,000,000,000,000,000,000

Source: [@kyrpides](https://twitter.com/@kyrpides)
Lack of Specialized Compute Capability

Specialized Machine for Sequencing

FAST

General-Purpose Machine for Analysis

SLOW
## Technology Dictates Algorithm Complexity

### Short Reads (Illumina)

<table>
<thead>
<tr>
<th>Step</th>
<th>Process</th>
<th>Library preparation</th>
<th>Sequencing</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sequencing</td>
<td>6.5 hours</td>
<td>68.2 Gb/hour</td>
</tr>
<tr>
<td>2</td>
<td>Basecalling</td>
<td>104.4 Gb/hour</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Quality Control</td>
<td>1339.2 Gb/hour</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Read Mapping</td>
<td>0.2 Gb/hour</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Variant Calling</td>
<td>1.2 Gb/hour</td>
<td></td>
</tr>
</tbody>
</table>

### Ultra-long Reads (ONT)

<table>
<thead>
<tr>
<th>Step</th>
<th>Process</th>
<th>Library preparation</th>
<th>Sequencing</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sequencing</td>
<td>24 hours</td>
<td>4.1 Gb/hour</td>
</tr>
<tr>
<td>2</td>
<td>Basecalling</td>
<td>0.833 Gb/hour</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Quality Control</td>
<td>3420 Gb/hour</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Read Mapping</td>
<td>1.7 Gb/hour</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Variant Calling</td>
<td>0.044 Gb/hour</td>
<td></td>
</tr>
</tbody>
</table>

### Accurate Long Reads (PacBio)

<table>
<thead>
<tr>
<th>Step</th>
<th>Process</th>
<th>Library preparation</th>
<th>Sequencing</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sequencing</td>
<td>24 hours</td>
<td>5.3 Gb/hour</td>
</tr>
<tr>
<td>2</td>
<td>Basecalling</td>
<td>8.3 Gb/hour</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Quality Control</td>
<td>1081 Gb/hour</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Read Mapping</td>
<td>1.4 Gb/hour</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Variant Calling</td>
<td>1.1 Gb/hour</td>
<td></td>
</tr>
</tbody>
</table>

---

Alser+,, *Going From Molecules to Genomic Variations to Scientific Discovery: Intelligent Algorithms and Architectures for Intelligent Genome Analysis*, arXiv 2022
Today’s Computing Systems

von Neumann model, 1945
where the CPU can access data stored in an off-chip main memory only through power-hungry bus

Storage (SSD/HDD) Main Memory Microprocessor

Burks, Goldstein, von Neumann, “Preliminary discussion of the logical design of an electronic computing instrument,” 1946.
Data analysis is performed far away from the data.
Data Movement Dominates Performance

- **Data movement** dominates performance and is a **major** system **energy bottleneck** (accounting for 40%-62%)

---

* Boroumand et al., “Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks,” ASPLOS 2018
* Kestor et al., “Quantifying the Energy Cost of Data Movement in Scientific Applications,” IISWC 2013
* Pandiyan and Wu, “Quantifying the energy cost of data movement for emerging smart phone workloads on mobile platforms,” IISWC 2014

---

Single memory request **consumes** >160x-800x more energy compared to performing an **addition operation**
Read Mapping Execution Time

>60% of the read mapper’s execution time is spent in sequence alignment

ONT FASTQ size: 103MB (151 reads), Mean length: 356,403 bp, std: 173,168 bp, longest length: 817,917 bp
Computational Cost is Mathematically Proven

[Submitted on 1 Dec 2014 (v1), last revised 15 Aug 2017 (this version, v4)]

**Edit Distance Cannot Be Computed in Strongly Subquadratic Time (unless SETH is false)**

Arturs Backurs, Piotr Indyk

The edit distance (a.k.a. the Levenshtein distance) between two strings is defined as the minimum number of insertions, deletions or substitutions of symbols needed to transform one string into another. The problem of computing the edit distance between two strings is a classical computational task, with a well-known algorithm based on dynamic programming. Unfortunately, all known algorithms for this problem run in nearly quadratic time.

In this paper we provide evidence that the near-quadratic running time bounds known for the problem of computing edit distance might be tight. Specifically, we show that, if the edit distance can be computed in time $O(n^{2-\delta})$ for some constant $\delta > 0$, then the satisfiability of conjunctive normal form formulas with $N$ variables and $M$ clauses can be solved in time $M^{O(1)2^{(1-\epsilon)N}}$ for a constant $\epsilon > 0$. The latter result would violate the Strong Exponential Time Hypothesis, which postulates that such algorithms do not exist.

https://arxiv.org/abs/1412.0348
Large Search Space for Mapping Location

98% of candidate locations have high dissimilarity with a given read

Xin et al, BMC genomics (2013)
Computing System

Leiserson+, "There's plenty of room at the Top: What will drive computer performance after Moore’s law?", Science, 2020

Richard Feynman, "There's Plenty of Room at the Bottom: An Invitation to Enter a New Field of Physics", a lecture given at Caltech, 1959.

Image source: https://science.sciencemag.org/content/368/6495/eaam9744
### Software & Hardware Optimizations

#### Multiplying Two 4096-by-4096 Matrices

```python
for i in xrange(4096):
    for j in xrange(4096):
        for k in xrange(4096):
            C[i][j] += A[i][k] * B[k][j]
```

<table>
<thead>
<tr>
<th>Implementation</th>
<th>Running time (s)</th>
<th>Absolute speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>Python</td>
<td>25,552.48</td>
<td>1x</td>
</tr>
<tr>
<td>Java</td>
<td>2,372.68</td>
<td>11x</td>
</tr>
<tr>
<td>C</td>
<td>542.67</td>
<td>47x</td>
</tr>
<tr>
<td>Parallel loops</td>
<td>69.80</td>
<td>366x</td>
</tr>
<tr>
<td>Parallel divide and conquer</td>
<td>3.80</td>
<td>6,727x</td>
</tr>
<tr>
<td>plus vectorization</td>
<td>1.10</td>
<td>23,224x</td>
</tr>
<tr>
<td>plus AVX intrinsics</td>
<td>0.41</td>
<td>62,806x</td>
</tr>
</tbody>
</table>

Leiserson+, "There’s plenty of room at the Top: What will drive computer performance after Moore’s law?", Science, 2020
# FASTQ Parsing

<table>
<thead>
<tr>
<th>Program</th>
<th>Language</th>
<th>t_{ gzip } (s)</th>
<th>t_{ plain } (s)</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>fqcnte_r2_needletail.rs</td>
<td>Rust</td>
<td>9.3</td>
<td>0.8</td>
<td>needletail; fasta/4-line fastq</td>
</tr>
<tr>
<td>fqcnte_c1_kseq.c</td>
<td>C</td>
<td>9.7</td>
<td>1.4</td>
<td>multi-line fasta/fastq</td>
</tr>
<tr>
<td>fqcnte_cr1_klib.cr</td>
<td>Crystal</td>
<td>9.7</td>
<td>1.5</td>
<td>kseq.h port</td>
</tr>
<tr>
<td>fqcnte_nim1_klib.nim</td>
<td>Nim</td>
<td>10.5</td>
<td>2.3</td>
<td>kseq.h port</td>
</tr>
<tr>
<td>fqcnte_jl1_klib.jl</td>
<td>Julia</td>
<td>11.2</td>
<td>2.9</td>
<td>kseq.h port</td>
</tr>
<tr>
<td>fqcnte_js1_k8.js</td>
<td>Javascript</td>
<td>17.5</td>
<td>9.4</td>
<td>kseq.h port</td>
</tr>
<tr>
<td>fqcnte_go1.go</td>
<td>Go</td>
<td>19.1</td>
<td>2.8</td>
<td>4-line only</td>
</tr>
<tr>
<td>fqcnte_lua1_klib.lua</td>
<td>LuaJIT</td>
<td>28.6</td>
<td>27.2</td>
<td>partial kseq.h port</td>
</tr>
<tr>
<td>fqcnte_py2_rfq.py</td>
<td>PyPy</td>
<td>28.9</td>
<td>14.6</td>
<td>partial kseq.h port</td>
</tr>
<tr>
<td>fqcnte_py2_rfq.py</td>
<td>Python</td>
<td>42.7</td>
<td>19.1</td>
<td>partial kseq.h port</td>
</tr>
</tbody>
</table>

[https://github.com/lh3/biofast](https://github.com/lh3/biofast)
We need intelligent algorithms and intelligent architectures that handle data well.
Several Genome Analysis Pipelines

Sequencing Technology:
- Illumina
- ONT
- PacBio (HiFi)

Species:
- E. Coli
- Human
- Yeast
- Zebra Fish
- Mice
- Fruit Fly

Reference Genomes

Sketching/ Indexing

Mapping
- Read Mapper:
  - BWA-MEM2
  - Minimap2
  - NGM-LR
  - Bowtie2

Assembly
- De novo Assembler (Long Reads):
  - Canu
  - Miniasm (uses Minimap2)
- De novo Assembler (Short Reads):
  - ABYSS
  - SPAdes (small genomes)

Variant Calling
- Variant Caller:
  - LuMPY
  - VariationHunter
  - GATK
  - TaRDiS
  - Freebayes
  - DELLY
  - Platypus
  - SAMtools
  - Genome STRiP

Polishing
- Assembly polisher:
  - Apollo
  - Racon
  - Pilon
  - Quiver (PB reads)
  - Arrow (PB reads, Not published yet)
  - NanoPolish (ONP reads)

Variant Caller:
- LuMPY
- VariationHunter
- GATK
- TaRDiS

Assembly Polishing
- Apollo
- Racon
- Pilon
- Quiver (PB reads)
- Arrow (PB reads, Not published yet)
- NanoPolish (ONP reads)

Taxonomy Profiling
- Kraken2
- Metalign
- MiCoP

Coverage:
- Low 2x - 30x
- Moderate 30x - 100x
- High >250x

Read Length:
- Short 100bp - 250bp
- Long 200bp – 2Mbp (>200bp)
- HiFi 10K-20Kbp

Basecalling

Read Set

Read Correction
- HALC
- LSC
- Hercules
- LoRDEC
- LoRMA
- Proovread
- ColorMap

De novo Assembler (Long Reads):
- Canu
- Miniasm (uses Minimap2)
De novo Assembler (Short Reads):
- ABYSS
- SPAdes (small genomes)
Accelerating Read Mapping

1. Indexing
   - Reference Genome
   - k-mers
   - Index
   - k-mer content: 1, 4, 6, 3, 5, 12, 50, 52
   - k-mer locations: 2, 100, 17, 19, 23, 90
   - Locating common k-mers: 1, 4, 6, 3, 5, 12, 17, 19

2. Pre-Alignment Filtering
   - Read
   - k-mers
   - Reference subsequences extracted at each common k-mer location

3. Sequence Alignment
   - Dynamic Programming (DP) Matrix
   - Output
   - SAM file (alignment score, edit distance, type and location of each edit)

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

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"

Our Contributions

Near-memory/In-memory Pre-alignment Filtering
- GRIM-Filter [BMC Genomics’18]
- SneakySnake [IEEE Micro’21]
- GenASM [MICRO 2020]

In-storage Sequence Alignment
- GenStore [ASPLOS 2022]

Near-memory Sequence Alignment
- GateKeeper [Bioinformatics’17]
- MAGNET [AACBB’18]
- Shouji [Bioinformatics’19]
- GateKeeper-GPU [arXiv’21]
- SneakySnake [Bioinformatics’20]

Specialized Pre-alignment Filtering Accelerators (GPU, FPGA)
- GRIM-Filter [BMC Genomics’18]
- SneakySnake [IEEE Micro’21]
- GenASM [MICRO 2020]

SeGrA [ISCA 2022]
Specialized Hardware for Pre-alignment Filtering

Mohammed Alser, Taha Shahroodi, Juan-Gomez Luna, Can Alkan, and Onur Mutlu, "SneakySnake: A Fast and Accurate Universal Genome Pre-Alignment Filter for CPUs, GPUs, and FPGAs" Bioinformatics, 2020.

[Source Code]
[Online link at Bioinformatics Journal]
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 incorrect mappings.
2. Preserve all correct mappings.
3. Do it quickly.
Key observation:
- Correct alignment is a sequence of non-overlapping long matches.

Dot plot, dot matrix (Lipman and Pearson, 1985)
**Key observation:**
- Correct alignment is a sequence of non-overlapping long matches

**Key idea:**
- Approximate edit distance calculation is similar to Single Net Routing problem in VLSI chip
Given two genomic sequences, a reference sequence $R[1 \ldots m]$ and a query sequence $Q[1 \ldots m]$, and an edit distance threshold $E$, we calculate the entry $Z[i, j]$ of the chip maze, where $1 \leq i \leq (2E + 1)$ and $1 \leq j \leq m$, as follows:

$$Z[i, j] = \begin{cases} 
0, & \text{if } i = E + 1, Q[j] = R[j], \\
0, & \text{if } 1 \leq i \leq E, Q[j - i] = R[j], \\
0, & \text{if } i > E + 1, Q[j + i - E - 1] = R[j], \\
1, & \text{otherwise}
\end{cases}$$

(1)

<table>
<thead>
<tr>
<th>column</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>3rd Upper Diagonal</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2nd Upper Diagonal</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1st Upper Diagonal</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Main Diagonal</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1st Lower Diagonal</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2nd Lower Diagonal</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3rd Lower Diagonal</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
SneakySnake Walkthrough

Building Neighborhood Map  Finding the Optimal Routing Path  Examining the Snake Survival

\[ E = 3 \]
SneakySnake Walkthrough

- Building Neighborhood Map
- Finding the Optimal Routing Path
- Examining the Snake Survival

Diagram showing a path from the entrance to the exit, with numbers indicating steps and a small image of a snake.
SneakySnake Walkthrough

This is what you actually need to **build** and it can be done **on-the-fly**!
FPGA Resource Analysis

- FPGA resource usage for a single filtering unit of GateKeeper, Shouji, and Snake-on-Chip for a sequence length of 100 and under different edit distance thresholds (E).

<table>
<thead>
<tr>
<th>E (bp)</th>
<th>GateKeeper</th>
<th>Shouji</th>
<th>Snake-on-Chip</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Slice LUT</td>
<td>Slice Register</td>
<td>No. of Filtering Units</td>
</tr>
<tr>
<td>2</td>
<td>0.39%</td>
<td>0.01%</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>0.71%</td>
<td>0.01%</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>0.69%</td>
<td>0.08%</td>
<td>16</td>
</tr>
<tr>
<td>2</td>
<td>1.72%</td>
<td>0.16%</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>1.42%</td>
<td>0.34%</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>0.68%</td>
<td>0.16%</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>1.42%</td>
<td>0.34%</td>
<td>16</td>
</tr>
</tbody>
</table>
Key Results of SneakySnake

- SneakySnake is up to **four orders of magnitude more accurate** than Shouji (Bioinformatics’19) and GateKeeper (Bioinformatics’17)

- Using short reads, SneakySnake **accelerates Edlib** (Bioinformatics’17) and Parasail (BMC Bioinformatics’16) by
  - up to $37.7 \times$ and $43.9 \times$ (>12× on average), on CPUs
  - up to $413 \times$ and $689 \times$ (>400× on average) with FPGA/GPU acceleration

- Using long reads, SneakySnake **accelerates Parasail and KSW2** by $140.1 \times$ and $17.1 \times$ on average, respectively, on CPUs
Can We Do Better?

Alleviating Data Movement Bottlenecks
We need to design mapping & filtering algorithms that fit processing-in-memory
Near-memory Pre-alignment Filtering

Gagandeep Singh, Mohammed Alser, Damla Senol Cali, Dionysios Diamantopoulos, Juan Gomez-Luna, Henk Corporaal, Onur Mutlu,
“FPGA-Based Near-Memory Acceleration of Modern Data-Intensive Applications”
[Source Code]
Near-memory SneakySnake

- Problem: Read Mapping is heavily bottlenecked by data movement from main memory

- Solution: Perform read mapping near where data resides (i.e., near-memory)

- We carefully redesigned the accelerator logic of SneakySnake to exploit near-memory computation capability on modern FPGA boards with high-bandwidth memory
Heterogeneous System: CPU+FPGA

We evaluate two POWER9+FPGA systems:

1. **HBM-based AD9H7 board**: Xilinx Virtex UltraScale+™ XCVU37P-2
2. **DDR4-based AD9V3 board**: Xilinx Virtex UltraScale+™ XCVU3P-2

FPGA + HBM on the same package substrate

POWER9 AC922

HBM-based AD9H7 board

Source: IBM

Source: AlphaData

DDR4-based AD9V3 board

Source: AlphaData

Source: AlphaData

Source: IBM
Near-memory pre-alignment filtering improves performance and energy efficiency by 27.4× and 133×, respectively, over a 16-core (64 hardware threads) IBM POWER9 CPU.
Near-memory Pre-alignment Filtering

Gagandeep Singh, Mohammed Alser, Damla Senol Cali, Dionysios Diamantopoulos, Juan Gomez-Luna, Henk Corporaal, Onur Mutlu,

“FPGA-Based Near-Memory Acceleration of Modern Data-Intensive Applications”
[Source Code]
Mohammed Alser, Taha Shahroodi, Juan Gómez Luna, Can Alkan, and Onur Mutlu, "SneakySnake: A Fast and Accurate Universal Genome Pre-Alignment Filter for CPUs, GPUs, and FPGAs" Bioinformatics, 2020.
[Source Code]
[Online link at Bioinformatics Journal]
GRIM-Filter

GRIM-Filter

- **Key observation:** FPGA and GPU accelerators are Heavily bottlenecked by Data Movement.

- **Key idea:** exploiting the high memory bandwidth and the logic layer of 3D-stacked memory to perform highly-parallel filtering in the DRAM chip itself.

- **Key results:**
  - We propose an algorithm called **GRIM-Filter**
  - GRIM-Filter with processing-in-memory is 1.8x-3.7x (2.1x on average) **faster than FastHASH filter** (BMC Genomics’13) across real data sets.
  - GRIM-Filter has 5.6x-6.4x (6.0x on average) lower falsely accepted pairs than **FastHASH filter** (BMC Genomics’13) across real data sets.
Each DRAM layer is organized as an array of banks
- A bank is an array of cells with a row buffer to transfer data
GRIM-Filter: Bitvectors

- Represent each bin with a **bitvector** that holds the occurrence of all permutations of a small string (**token**) in the bin.

- To account for matches that straddle bins, we employ **overlapping** bins.
  - A read will now always completely fall within a single bin.
GRIM-Filter: Bitvectors

Storing all bitvectors requires $4^n \times t$ bits in memory, where $t =$ number of bins & $n =$ token length.

For bin size $\sim 200$, and $n = 5$, memory footprint $\sim 3.8 \text{ GB}$
Each DRAM layer is organized as an array of **banks**
- A **bank** is an array of cells with a row buffer to transfer data

- The layout of bitvectors in a bank enables filtering many bins in parallel
More on GRIM-Filter

Background: Read Mappers

We now have **sequenced reads** and want a **full genome** via Read Mapping.

We map **reads** to a known **reference genome** (>99.9% similarity across humans) with some minor errors allowed.

Because of high similarity, long sequences in **reads** perfectly match in the **reference genome**.

```
GACTGTGTCAA
```

We can use a hash table to help quickly map the **reads**.

Livestream - P&S Accelerating Genome Analysis with FPGAs, GPUs, and New Execution Paradigms (Fall 2021)

**GRIM-Filter: Fast Seed Location Filtering in DNA Read Mapping w/ Processing-in-Memory**

- Jeremie Kim

https://www.youtube.com/watch?v=j5-I84iNVd8
More on GRIM-Filter

Proceedings of the 16th Asia Pacific Bioinformatics Conference (APBC), Yokohama, Japan, January 2018.
arxiv.org Version (pdf)
GenASM Framework [MICRO 2020]

- Damla Senol Cali, Gurpreet S. Kalsi, Zulal Bingol, Can Firtina, Lavanya Subramanian, Jeremie S. Kim, Rachata Ausavarungnirun, Mohammed Alser, Juan Gomez-Luna, Amirali Boroumand, Anant Nori, Allison Scibisz, Sreenivas Subramoney, Can Alkan, Saugata Ghose, and Onur Mutlu,

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


[Lightning Talk Video (1.5 minutes)]
[Lightning Talk Slides (pptx) (pdf)]
[Talk Video (18 minutes)]
[Slides (pptx) (pdf)]
Near-memory GenASM Framework

- **Our goal:** Accelerate approximate string matching (ASM) by designing a fast and flexible framework, which can accelerate multiple steps of genome sequence analysis.

- **Key ideas:** Exploit the high memory bandwidth and the logic layer of 3D-stacked memory to perform highly-parallel ASM in the DRAM chip itself.

- Modify and extend Bitap\(^1,2\), ASM algorithm with fast and simple bitwise operations, such that it now:
  - Supports long reads
  - Supports traceback
  - Is highly parallelizable

- **Co-design** of our modified scalable and memory-efficient algorithms with low-power and area-efficient hardware accelerators

---

# Key Results of the GenASMs Framework

<table>
<thead>
<tr>
<th>Result</th>
<th>Speedup</th>
<th>Power Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>(1) Read Alignment</strong></td>
<td>116×</td>
<td>37×</td>
</tr>
<tr>
<td>111×</td>
<td>33×</td>
<td>Minimap2 (state-of-the-art SW)</td>
</tr>
<tr>
<td>3.9×</td>
<td>2.7×</td>
<td>BWA-MEM (state-of-the-art SW)</td>
</tr>
<tr>
<td>1.9×</td>
<td>82%</td>
<td>Darwin (state-of-the-art HW)</td>
</tr>
<tr>
<td><strong>(2) Pre-Alignment Filtering</strong></td>
<td>3.7×</td>
<td>1.7×</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shouji (state-of-the-art HW)</td>
</tr>
<tr>
<td><strong>(3) Edit Distance Calculation</strong></td>
<td>22–12501×</td>
<td>548–582×</td>
</tr>
<tr>
<td>9.3–400×</td>
<td>67×</td>
<td>Edlib (state-of-the-art SW)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ASAP (state-of-the-art HW)</td>
</tr>
</tbody>
</table>
More on GenASM

Livestream - P&S Accelerating Genome Analysis with FPGAs, GPUs, and New Execution Paradigms (Fall 2021)
Comp. Architecture - Lecture 9a: GenASM: Approx. String Matching Accelerator (ETH Zürich, Fall 2020)

https://www.youtube.com/watch?v=XoLpzmN-Pas
More on Optimizing GenASM

P&S Heterogeneous Systems
Algorithmic Improvement and GPU Acceleration of the GenASM Algorithm

Joël Lindegger
ETH Zürich
Spring 2022
24 June 2022

HetSys Course: Lecture 16: GPU Acceleration of Genome Sequence Alignment (Fall 2022)

https://www.youtube.com/watch?v=pEIoUxlqBGY
Computing is Still Bottlenecked by Data Movement from storage to the rest of the system
GenStore (ASPLOS 2022)

Nika Mansouiri Ghiasi, Jisung Park, Harun Mustafa, Jeremie Kim, Ataberk Olgun, Arvid Gollwitzer, Damla Senol Cali, Can Firtina, Haiyu Mao, Nour Almadhoun Alserr, Rachata Ausavarungrun, Nandita Vijaykumar, **Mohammed Alser**, Onur Mutlu

"GenStore: A High-Performance and Energy-Efficient In-Storage Computing System for Genome Sequence Analysis",

ASPLOS 2022

**GenStore: A High-Performance In-Storage Processing System for Genome Sequence Analysis**

<table>
<thead>
<tr>
<th>Nika Mansouiri Ghiasi</th>
<th>Jisung Park</th>
<th>Harun Mustafa</th>
<th>Jeremie Kim</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETH Zürich</td>
<td>ETH Zürich</td>
<td>ETH Zürich</td>
<td>ETH Zürich</td>
</tr>
<tr>
<td>Switzerland</td>
<td>Switzerland</td>
<td>Switzerland</td>
<td>Switzerland</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ataberk Olgun</th>
<th>Arvid Gollwitzer</th>
<th>Damla Senol Cali</th>
<th>Can Firtina</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETH Zürich</td>
<td>ETH Zürich</td>
<td>Bionano Genomics</td>
<td>ETH Zürich</td>
</tr>
<tr>
<td>Switzerland</td>
<td>Switzerland</td>
<td>USA</td>
<td>Switzerland</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Haiyu Mao</th>
<th>Nour Almadhoun Alserr</th>
<th>Rachata Ausavarungrunirun</th>
<th>Nandita Vijaykumar</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETH Zürich</td>
<td>ETH Zürich</td>
<td>KMUTNB</td>
<td>University of Toronto</td>
</tr>
<tr>
<td>Switzerland</td>
<td>Switzerland</td>
<td>Thailand</td>
<td>Canada</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mohammed Alser</th>
<th>Onur Mutlu</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETH Zürich</td>
<td>ETH Zürich</td>
</tr>
<tr>
<td>Switzerland</td>
<td>Switzerland</td>
</tr>
</tbody>
</table>
Key Ideas of GenStore (ASPLOS 2022)

**GenStore-EM (exactly-matching reads filter):** In some cases, a large fraction of reads **exactly match** to subsequences of the reference genome.

**GenStore-NM (non-matching reads filter):** In some cases, a large fraction of reads **do not match** to subsequences of the reference genome.

GenStore-EM: 2.1-6.1× speedup & 3.92x energy saving compared to minimap2.
GenStore-NM: 1.4-33.6x speedup & 27.17x energy saving compared to minimap2.
More on GenStore (ASPLOS 2022)

GenStore: A High-Performance In-Storage Processing System for Genome Analysis – ASPLOS'22 Talk

https://www.youtube.com/watch?app=desktop&v=bv7hgXOOMjk
Haiyu Mao, Mohammed Alser, Mohammad Sadrosadati, Can Firtina, Akanksha Baranwal, Damla Senol Cali, Aditya Manglik, Nour Almadhoun Alserr, Onur Mutlu
“GenPIP: In-Memory Acceleration of Genome Analysis via Tight Integration of Basecalling and Read Mapping”
Innovations Require Change

- GenPIP processes reads at the granularity of a chunk instead of the complete read sequence, increasing parallelism and resource utilization by overlapping the execution of different steps.

GenPIP provides 41.6x and 8.4x speedup and 32.8x and 20.8x energy reduction compared to CPU and GPU state-of-the-art solutions.
More on GenPIP (MICRO 2022)

Limitation 2: Wasted Computation

Using a human dataset in [NC’19] as an example:

- 100% Raw Signals
- 79.5% Basecalling
- 69.5% Reads
- 100% Read quality control
- 79.5% High-quality reads
- 69.5% Read mapping
- 100% Mapped reads

- 20.5% Low-quality reads
- 10% Unmapped reads

A considerable amount of computation on **useless data** due to
- Low-quality reads
- Unmapped reads

https://www.youtube.com/watch?v=PWWBtrL60dQ&t=8290s
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 (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](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.timelogic.com/catalog/775)
- [http://www.bcgsc.ca/platform/bioinfo/software/XpressAlign/releases/1.0](http://www.bcgsc.ca/platform/bioinfo/software/XpressAlign/releases/1.0)
- [https://www.sevenbridges.com/amazon/](https://www.sevenbridges.com/amazon/)
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](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.
Dream
and, they will come

Computing landscape is very different from 10-20 years ago
Illumina DRAGEN Bio-IT Platform (2018)

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

[Image of FPGA board(s)]

[Links]
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

GPU board(s)

PERFORMANCE COMPARISON
Germline End-to-End Secondary Analysis

1,200 minutes

<table>
<thead>
<tr>
<th></th>
<th>CPU/GATK</th>
<th>8X T4</th>
<th>8X V100</th>
<th>8X A100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>52 minutes</td>
<td>35 minutes</td>
<td>23 minutes</td>
<td></td>
</tr>
</tbody>
</table>

https://developer.nvidia.com/clara-parabricks
Moving Forward
Cerebras’s Wafer Scale Engine (2019)

- The largest ML accelerator chip
- 400,000 cores

Cerebras WSE
1.2 Trillion transistors
46,225 mm$^2$

Largest GPU
21.1 Billion transistors
815 mm$^2$

https://www.cerebras.net/cerebras-wafer-scale-engine-why-we-need-big-chips-for-deep-learning/
**TESLA Full Self-Driving Computer (2019)**

- ML accelerator: 260 mm$^2$, 6 billion transistors, 600 GFLOPS GPU, 12 ARM 2.2 GHz CPUs.
- Two redundant chips for better safety.

https://youtu.be/Ucp0TTmvqOE?t=4236
NextSeq 2000 with Analysis Capability

NextSeq 1000/2000 Integrates DRAGEN Bio-IT Platform On-Board

**DRAGEN Bio-IT platform:**
- Fast
- Accurate
- Industry standard pipelines
- For both novice and expert users

**Pipelines available on-board:**
- DRAGEN Enrichment pipeline
- DRAGEN RNA pipeline
- DRAGEN Germline
- DRAGEN Single Cell RNA
- Generate FASTQ via BCL Convert
- Additional pipelines available in BaseSpace Sequence Hub

For Research Use Only. Not for use in diagnostic procedures.
NVIDIA is claiming a **7x improvement** in dynamic programming algorithm (**DPX instructions**) performance on a single H100 versus naïve execution on an A100.

The vision of BioPIM is the realization of **cheap, ultra-fast and ultra-low energy mobile genomics** that eliminates the current dependence of sequence analysis on large and power-hungry computing clusters/data-centers.
UPMEM Processing-in-DRAM Engine (2019)

- **Processing in DRAM Engine**

  - **Includes** standard DIMM modules, with a large number of DPU processors combined with DRAM chips.

- **Replaces** standard DIMMs
  - DDR4 R-DIMM modules
    - 8GB+128 DPUs (16 PIM chips)
    - Standard 2x-nm DRAM process
  - **Large amounts of** compute & memory bandwidth

---


Onur Mutlu, Computer Architecture Lecture 2b, Fall 2019, ETH Zurich
Where is Read Mapping Going Next?

NO

machine gives the complete sequence of genome as output

What if we have such a machine?!

Think about comparing large genomes together as in whole-genome alignment, metagenomics, pangenomics, ...

SAFARI
Open Questions

How and where to enable fast, accurate, cheap, privacy-preserving, and exabyte-scale analysis of genomic data?
Processing Genomic Data Where it Makes Sense

Modern systems

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.
Near-memory Pre-alignment Filtering

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

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

Read Mapping in 111 pages!

In-depth analysis of 107 read mappers (1988-2020)

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]
Feedback From Our Community!

James Ferguson
@Psy_Fer_
This is awesome! I've got my evening reading sorted.

Stéphane Le Crom
@slecrom
Very complete article on the evolution of read alignment algorithms. #NGS #genomics

Svetlana Gorokhova
@SGorokhova
An impressive overview of read alignment methods over the last three decades

BContrerasMoreira @BrunoContrerasM · Sep 10
Replying to @mealser @GenomeBiology and 3 others
Buen hilo de repaso sobre la evolución de los algoritmos de alineamiento de secuencias a medida que ha mejorado la tecnología de secuenciación

https://twitter.com/mealser/status/1435223377644503040
More on Accelerating Genome Analysis ...

- Mohammed Alser,
  "Accelerating Genome Analysis: A Primer on an Ongoing Journey"
  Talk at RECOMB 2021, Virtual, August 30, 2021.
  [Slides (pptx) (pdf)]
  [Talk Video (27 minutes)]
  [Related Invited Paper (at IEEE Micro, 2020)]
More on Intelligent Genome Analysis …

- Mohammed Alser,
  "Computer Architecture - Lecture 10: Intelligent Genome Analysis"
  ETH Zurich, Computer Architecture Course, Fall 2021, Lecture 10, Virtual, 29 October 2021.
  [Slides (pptx) (pdf)]
  [Talk Video (3 hour 2 minutes, including Q&A)]
  [Related Invited Paper (at IEEE Micro, 2020)]
More on Intelligent Genome Analysis …

- Mohammed Alser,
  "Computer Architecture - Lecture 8: Intelligent Genome Analysis"
  ETH Zurich, Computer Architecture Course, Lecture 8, Virtual, 15 October 2021.
  [Slides (pptx) (pdf)]
  [Talk Video (2 hour 54 minutes, including Q&A)]
  [Related Invited Paper (at IEEE Micro, 2020)]
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)]
Two P&S Genomics Courses

Livestream - P&S Genome Sequencing on Mobile Devices (Fall 2021)

9 videos • 75 views • Updated 5 days ago

https://www.youtube.com/playlist?list=PL5Q2soXY2Zi_U2F8yrrNPD9CjcM6CFQXv
## Course Materials

### 2021 Meetings/Schedule

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Livestream</th>
<th>Meeting</th>
<th>Learning Materials</th>
<th>Assignments</th>
</tr>
</thead>
<tbody>
<tr>
<td>W1</td>
<td>5.10 Tue.</td>
<td>Live YouTube</td>
<td><strong>M1: P&amp;S Accelerating Genomics</strong>&lt;br&gt;Course Introduction &amp; Project Proposals&lt;br&gt;(PDF) (PPT)&lt;br&gt;YouTube Video</td>
<td>Requi...</td>
<td>Required Materials Recommended Materials</td>
</tr>
<tr>
<td>W2</td>
<td>20.10 Wed.</td>
<td>Live YouTube</td>
<td><strong>M2: Introduction to Sequencing</strong>&lt;br&gt;(PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W3</td>
<td>27.10 Wed.</td>
<td>Live YouTube</td>
<td><strong>M3: Read Mapping</strong>&lt;br&gt;(PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W4</td>
<td>3.11 Wed.</td>
<td>Live YouTube</td>
<td><strong>M4: GateKeeper</strong>&lt;br&gt;(PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W5</td>
<td>10.11 Wed.</td>
<td>Live YouTube</td>
<td><strong>M5: MAGNET &amp; Shouji</strong>&lt;br&gt;(PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W6</td>
<td>17.11 Wed.</td>
<td></td>
<td><strong>M6.1: SneakySnake</strong>&lt;br&gt;(PDF) (PPT) Video</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>M6.2: GRIM-Filter</strong>&lt;br&gt;(PDF) (PPT) YouTube Video</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W7</td>
<td>24.11 Wed.</td>
<td></td>
<td><strong>M7: GenASM</strong>&lt;br&gt;(PDF) (PPT) YouTube Video</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Detailed Lectures on Genome Analysis

- **Computer Architecture, Fall 2020, Lecture 3a**
  - **Introduction to Genome Sequence Analysis** (ETH Zürich, Fall 2020)
  - [Link](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)
  - [Link](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)
  - [Link](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)
  - [Link](https://www.youtube.com/watch?v=rgjl8ZyLsAg&list=PL5Q2soXY2Zi9E2bBVAgCgLgwIDRQDTyId)

---

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


Prior Research on Genome Analysis (2/2)


Seminar in Computer Architecture
Intelligent Genomic Analyses

Dr. Mohammed Alser
@mealser

ETH Zurich
Spring 2023
2 March 2023
Backup Slides
**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)
Mechanisms

- **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)

The figure illustrates the calculation of the Shifted Hamming Distance between two strings, ISTANBUL and ISTANBUL. The process involves:

1. **XOR** operation on corresponding characters of the two strings:
   - XOR of ISTANBUL and ISTANBUL results in 0 0 0 1 1 1 1.

2. **AND** operation on the result of the XOR:
   - AND of 0 0 0 1 1 1 1 and ISTANBUL results in 1 1 1 0 0 0 0.

3. **Count 1's** in the result of the AND:
   - The count of 1's is 7 matches and 1 mismatch.

The final result indicates that there is a single deletion to transform ISTANBUL into ISTANBUL.
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”: 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>A</th>
<th>T</th>
<th>A</th>
<th>C</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<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>-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>-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>
<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>
<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>
<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>
<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>A</th>
<th>T</th>
<th>A</th>
<th>C</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<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>A</td>
<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>C</td>
<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>T</td>
<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>A</td>
<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>T</td>
<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>
<tr>
<td>A</td>
<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>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></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) * (ReadLength) 2-XOR operations.

And all masks, ACCEPT iff number of ‘1’ ≤ Threshold

- (2E)*(ReadLength) 2-AND operations.
- (ReadLength/4) 5-input LUT.
- $log_2$ ReadLength-bit counter.

Hamming mask

0100100011010001010110011100101010010

5-input LUT

Hamming mask after amending

011110001110001111111111111111111100011110

- (2E+1)*(ReadLength) 5-input LUT.
Configurable logic blocks (CLBs) are the main logic resources for implementing sequential as well as combinatorial circuits.
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(^{(1)})</th>
<th>Shift Registers(^{(1)})</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(^a)</td>
</tr>
<tr>
<td>Slice Register(^b)</td>
</tr>
</tbody>
</table>

\(^a\)LUT: look-up tables.
\(^b\)Flip-flop.
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 |
|-------------------------------------------------|-----------------|-----------------|
| Read length | 100 bp | 300 bp |
| Edit distance | 16 GateKeeper cores | 8 GateKeeper cores |
| Slice LUT | 32% | 50% |
| Slice register | 2% | 17% |
| Block memory | 2% | 2% |
GateKeeper Accelerator Architecture

- **Maximum data throughput** = \(~13.3\) 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
The Effect of Pre-Alignment (Theoretically)

Pre-alignment saves more than 40% to 80% of the total processing time

Filter+ Alignment

Total processing time without pre-alignment (sec)
Total processing time with pre-alignment (sec)
Ideal processing time for 90% pre-alignment rejection percentage

assumining alignment processes 100 Mappings/sec

Processing time (sec) for 1 million mappings

Target

Pre-alignment rejected mapping percentage and speed compared to alignment step
Filtering Accuracy

Fig. 10: The execution time of SneakySnake, Parasail, and SneakySnake integrated with Parasail using long sequences, (a) 10Kbp and (b) 100Kbp, and 40 CPU threads. The left y-axes of (a) and (b) are on a logarithmic scale. For each edit distance threshold value, we provide in the right y-axes of (a) and (b) the rate of accepted pairs (out of 100,000 pairs for 10Kbp and out of 74,687 pairs for 100Kbp) by SneakySnake that are passed to Parasail. We present the end-to-end speedup values obtained by integrating SneakySnake with Parasail.
**Long Read Mapping (SneakySnake vs KSW2)**

Fig. 11: The execution time of SneakySnake, KSW2, and SneakySnake integrated with KSW2 using long sequences, (a) 10Kbp and (b) 100Kbp, and a single CPU thread. The left y-axes of (a) and (b) are on a logarithmic scale. For each edit distance threshold value, we provide in the right y-axes of (a) and (b) the rate of accepted pairs (out of 100,000 pairs for 10Kbp and out of 74,687 pairs for 100Kbp) by SneakySnake that are passed to KSW2. We present the end-to-end speedup values obtained by integrating SneakySnake with KSW2.
Ongoing Directions

- **Seed Filtering Technique:**
  - **Goal:** Reducing the number of seed (k-mer) locations.
    - Heuristic (limits the number of mapping locations for each seed).
    - Supports exact matches only.

- **Pre-alignment Filtering Technique:**
  - **Goal:** Reducing the number of invalid mappings (>E).
    - Supports both exact and inexact matches.
    - Provides some falsely-accepted mappings.

- **Read Alignment Acceleration:**
  - **Goal:** Performing read alignment at scale.
    - Limits the numeric range of each cell in the DP table and hence supports limited scoring function.
    - May not support backtracking step due to random memory accesses.
The Need for Speed

Did we realize the need for faster genome analysis?

Alser+, "Technology dictates algorithms: Recent developments in read alignment", arXiv, 2020
The Need for Speed

- **High**
  - (Urgent) clinical seq.
  - Diagnosis
  - Treatment guidance
  - Infection control
  - Species/subspecies
  - Antibiotic resistance
  - Virus profiling
  - Coinfection
  - …

- **Moderate**
  - Other clinical
    - Tumor profiling
    - Cancer subtypes
    - Drug resistance
    - Rare disease diag.
  - Research
    - Genotype/phenotype
    - Causal mutations
    - Population genomics
    - Evolutionary biology
    - *De novo* assembly
    - …

- HPC, accelerators, embedded devices…

- Cloud, clusters, HPC, advanced workstations…
Did we Achieve Our Goal?

- Our goal is to **significantly reduce** the time spent on **calculating the optimal alignment** in genome analysis from hours to mere seconds using both **new algorithms & hardware accelerators**, given **limited computational resources** (i.e., personal computer or small hardware).
What is Intelligent Genome Analysis?

- Fast genome analysis
  - Real-time analysis

- Using intelligent architectures
  - Specialized HW with less data movement

- DNA is a valuable asset
  - Controlled-access analysis

- Population-scale genome analysis
  - Sequence anywhere at large scale!

- Avoiding erroneous analysis
  - E.g., your father is not your father

Bandwidth
Energy-efficiency & Latency
Privacy
Scalability
Accuracy
Where is Read Mapping Going Next?

African pan-genome contains ~10% more DNA than the current human reference genome.