P&S Mobile Genomics
Lecture 3: Read Mapping

Dr. Mohammed Alser
@mealser

ETH Zurich
Spring 2022
22 March 2022
Agenda for Lecture 2

- What is Genome Analysis?
- What is Intelligent Genome Analysis?
- How we Analyze Genome?
Agenda for Today

- What is Read Mapping?
- What Makes Read Mapper Slow?
- Algorithmic & Hardware Acceleration
  - Seed Filtering Technique
  - Pre-alignment Filtering Technique
  - Read Alignment Acceleration
Agenda for Today

- What is Read Mapping?
- What Makes Read Mapper Slow?
- Algorithmic & Hardware Acceleration
  - Seed Filtering Technique
  - Pre-alignment Filtering Technique
  - Read Alignment Acceleration
Map **reads** to a known reference genome with some minor differences allowed

DNA Sample “chemical format” → Reads “text format” → Reference genome “text format”
Solving the Puzzle

Reference genome

Reads

Cracking the 1st Human Genome Sequence

1990-2003: The Human Genome Project (HGP) provides a complete and accurate sequence of all DNA base pairs that make up the human genome and finds 20,000 to 25,000 human genes.

3.2 x10^9 bases
13 years
>3x10^9 $
Three Decades & Yet to be Complete!

The complete sequence of a human genome


doi: https://doi.org/10.1101/2021.05.26.445798

COMPLETING THE HUMAN GENOME

Researchers have been filling in incompletely sequenced parts of the human reference genome for 20 years, and have now almost finished it, with 3.05 billion DNA base pairs.

200 million new bases

0.3% of sequence might still have errors. Includes X but not Y chromosome. Count excludes mitochondrial DNA.
Obtaining the Human Reference Genome

- **GRCh38.p13**
- **Description:** Genome Reference Consortium Human Build 38 patch release 13 (GRCh38.p13)
- **Organism name:** [Homo sapiens (human)](https://www.ncbi.nlm.nih.gov/assembly/GCF_000001405.39)
- **Date:** 2019/02/28
- **3,099,706,404 bases**
- **Compressed .fna file (964.9 MB)**
How Long is DNA?

- 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
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
- PRJNA33847 • 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

<table>
<thead>
<tr>
<th>Run</th>
<th># of Spots</th>
<th># of Bases</th>
<th>Size</th>
<th>Published</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR240727</td>
<td>4,093,747</td>
<td>818.7M</td>
<td>387.2Mb</td>
<td>2013-03-22</td>
</tr>
</tbody>
</table>
Let’s learn how to map a read
Read Mapping: A Brute Force Algorithm

Reference

Read

Very expensive!

\[ O(m^2kn) \]

\( m \): read length

\( k \): no. of reads

\( n \): reference genome length
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

BConrrerasMoreira @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
Mapping a read is similar to querying the yellow pages!
Similar to Searching Yellow Pages!

- Step 1: Get the page number from the book’s index using a small portion of the name (e.g., 1st letter).
- Step 2: Retrieve the page(s).
- Step 3: Match the full name & get the phone number.
Matching Each Read with Reference Genome

.FASTA file:

>`NG_008679.1:5001-38170 Homo sapiens paired box 6 (PAX6)
ACCTTCATTTGACATTTAATCTCTGGGCAGGGAACGCAGCTGTCAAGATCT
GCCACTTCCCCCTGGGAGCGCGGTAGATGTTGGAACCGAGCTCTCACTGCCTCCTCCCG
CCTCCGCTCCAGTAAACCGCGCCGCTCGGGGCAGGGGGCTTCGCTGCTGCTTCAAAATCC
CCACGAGCTGTGCTCCCAAAATCAAGCAGGCAGGAGATGAAAGGGGGTGGAGGAGGGACTTGCTT
TCCCGAGTGTAATAGAGTACGTAATGAGGATTAATAGGAATACGGGAGCTGGTGGATCT
GAGCTGGAGTAGGGGGCGGAGTGCTGTGCTGTGCTGTCAAAAGCAGCTCGGACGCGAGAAAATGCA
GGAGTTGGGACGCACTTTTGCATCCAGACCTCTCTCTGCATCGCAGTTAACGCTTTGAAG
TCCGTACCCGCGCCCTGTGGAAAGCACACCCTGGCGGCTGGGACGTTGCAGCAGAAAGTTTCCC
GCGGTTGCAAAGATGCGATGTGGTGGAGCTGACCGAACAAGTCTAGAGATGGGGTTCTTCTCAGAAAGACGC

.FASTQ file:

>HWI-EAS209_0006_FC706VJ:5:58:5894:21141#ATCACG/1
TAGATCTTTTAGATNAG
+HWI-EAS209_0006_FC706VJ:5:58:5894:21141#ATCACG/1
efcfffeffcfefeffcfefefffddfe'deef"]_Ba^___[YBBBBBBBBBBRTT
Step 1: Indexing the Reference Genome
Hashing is the most popular indexing technique for read mapping since 1988

Alser+, "Technology dictates algorithms: Recent developments in read alignment", Genome Biology, 2021
Step 1: Indexing the Reference Genome

Index the first seed at location 1

Seed = k-mer (string of length k)

Index location list:

- Location 1: 1, 9, 16, 30
- Location 2: 2, 7, 60
- Location 3: 3, 5, 12
- Location 4: 4, 10, 18, 32
- Location 6: 6, 14

Seed location at the reference genome.
Genome Index Properties

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

- **Seeds** can be overlapping, non-overlapping, spaced, adjacent, 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
Performance of Human Genome Indexing

Alser+, "Technology dictates algorithms: Recent developments in read alignment", Genome Biology, 2021
Step 2: Query the Index Using Read Seeds
Step 2: Query the Index Using Read Seeds

read 1: CCTAGTATATTACGT

read 2: TATTCTTACCGTACTAGTAGTACGCCC

read 3: GCCTCTATATCGCTACTATAGTG

seed

location list

1 9 16 30

2 7 60

3 5 12

4 10 18 32

6 14

seed location at the reference genome

reference genome

location list from index data structure

1 9 16 30

2 7 60

3 5 12
We can query the Hash table with substrings from reads to quickly find a list of possible mapping locations.
Step 3: Sequence Alignment (Verification)

![Sequence Alignment Diagram]

The diagram shows a matrix representing sequence alignment. Each element in the matrix corresponds to a match or mismatch score between sequences. The text explains that a .bam/.sam file contains necessary alignment information (e.g., type, location, and number of each edit).
Step 3: Sequence Alignment (Verification)

- **Edit distance** is defined as the minimum number of edits (i.e. insertions, deletions, or substitutions) needed to make the read exactly match the reference segment.

![Sequence Alignment Diagram]

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

- **organization x translation**
  - Reference: organization
  - Read: translation
  - Edit distance = 4
Popular Algorithms for Sequence Alignment

Smith-Waterman remains the most popular algorithm since 1988.

Hamming distance is the second most popular technique since 2008.

Alser+, "Technology dictates algorithms: Recent developments in read alignment", Genome Biology, 2021
An Example of Hash Table Based Mappers

- Guaranteed to find all mappings → very sensitive
- Can tolerate up to $e$ errors

https://github.com/BilkentCompGen/mrfast

Personalized copy number and segmental duplication maps using next-generation sequencing

Can Alkan$^{1,2}$, Jeffrey M Kidd$^1$, Tomas Marques-Bonet$^{1,3}$, Gozde Aksay$^1$, Francesca Antonacci$^1$, Fereydoun Hormozdiari$^4$, Jacob O Kitzman$^1$, Carl Baker$^1$, Maika Malig$^1$, Onur Mutlu$^5$, S Cenk Sahinalp$^4$, Richard A Gibbs$^6$ & Evan E Eichler$^{1,2}$

Performance of Read Mapping

Alser+, "Technology dictates algorithms: Recent developments in read alignment", Genome Biology, 2021
The Need for Speed

Did we realize the **need for faster** genome analysis?

Alser+, "Technology dictates algorithms: Recent developments in read alignment", Genome Biology, 2021
Read Mapping

Map reads to a known reference genome with some minor differences allowed.
Metagenomics Analysis

Reads from different **unknown** donors at sequencing time are mapped to **many known reference** genomes.

genetic material recovered directly from environmental samples

Reads "text format"

Reference Database
Genomics vs. Metagenomics
Check Also CAMI II Paper

F. Meyer, A. Fritz, Z.L. Deng, D. Koslicki, A. Gurevich, G. Robertson, Mohammed Alser, and others

“Critical Assessment of Metagenome Interpretation - the second round of challenges”

bioRxiv, 2021
[Source Code]

doi: https://doi.org/10.1101/2021.07.12.451567
Check Also MiCoP

Nathan LaPierre, Serghei Mangul, Mohammed Alser, Igor Mandric, Nicholas C. Wu, David Koslicki & Eleazar Eskin

“MiCoP: microbial community profiling method for detecting viral and fungal organisms in metagenomic samples”

BMC Genomics, June 2019.

[Source code]
Challenges in Read Mapping

- Need to find many **mappings** of each read
- Need to **tolerate variances/sequencing errors** in each read
- Need to **map** each read **very fast** (i.e., performance is important, life critical in some cases)
- Need to **map** reads to both **forward and reverse strands**
Sequencing Technology:
• Illumina
• ONT
• PacBio (HiFi)

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

Reference Genomes

Basecalling

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

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

Reference Genomes

Read Corrector:
• HALC
• LSC
• Hercules
• LoRDEC
• LoRMA
• Proovread
• ColorMap

Mapping

Mapping

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

Variant Calling

Variant Caller:
• LuMPY
• VariationHunter
• GATK
• TaRDiS

Variant Caller:
• Freebayes
• DELLY
• Platypus
• SAMtools
• Genome STRiP

Polishing

Assembly

De novo Assembler (Long Reads):
• Canu
• Miniasm (uses Minimap2)

De novo Assembler (Short Reads):
• ABYSS
• SPAdes (small genomes)

Assembly Polisher:
• Apollo
• Racon
• Pilon
• Quiver (PB reads)
• Arrow (PB reads, Not published yet)
• NanoPolish (ONP reads)

Variant Calling

Taxonomy Profiling

De novo Assembler (Long Reads):
• Canu
• Miniasm (uses Minimap2)

De novo Assembler (Short Reads):
• ABYSS
• SPAdes (small genomes)

Assembly Polisher:
• Apollo
• Racon
• Pilon
• Quiver (PB reads)
• Arrow (PB reads, Not published yet)
• NanoPolish (ONP reads)

Variant Calling

Basecalling

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

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

Reference Genomes

Read Corrector:
• HALC
• LSC
• Hercules
• LoRDEC
• LoRMA
• Proovread
• ColorMap

Mapping

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

Variant Calling

Variant Caller:
• LuMPY
• VariationHunter
• GATK
• TaRDiS

Variant Caller:
• Freebayes
• DELLY
• Platypus
• SAMtools
• Genome STRiP

Polishing

Assembly

De novo Assembler (Long Reads):
• Canu
• Miniasm (uses Minimap2)

De novo Assembler (Short Reads):
• ABYSS
• SPAdes (small genomes)

Assembly Polisher:
• Apollo
• Racon
• Pilon
• Quiver (PB reads)
• Arrow (PB reads, Not published yet)
• NanoPolish (ONP reads)

Variant Calling

Basecalling

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

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

Reference Genomes

Read Corrector:
• HALC
• LSC
• Hercules
• LoRDEC
• LoRMA
• Proovread
• ColorMap

Mapping

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

Variant Calling

Variant Caller:
• LuMPY
• VariationHunter
• GATK
• TaRDiS

Variant Caller:
• Freebayes
• DELLY
• Platypus
• SAMtools
• Genome STRiP

Polishing

Assembly

De novo Assembler (Long Reads):
• Canu
• Miniasm (uses Minimap2)

De novo Assembler (Short Reads):
• ABYSS
• SPAdes (small genomes)

Assembly Polisher:
• Apollo
• Racon
• Pilon
• Quiver (PB reads)
• Arrow (PB reads, Not published yet)
• NanoPolish (ONP reads)

Variant Calling

Basecalling

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

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

Reference Genomes

Read Corrector:
• HALC
• LSC
• Hercules
• LoRDEC
• LoRMA
• Proovread
• ColorMap

Mapping

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

Variant Calling

Variant Caller:
• LuMPY
• VariationHunter
• GATK
• TaRDiS

Variant Caller:
• Freebayes
• DELLY
• Platypus
• SAMtools
• Genome STRiP

Polishing

Assembly

De novo Assembler (Long Reads):
• Canu
• Miniasm (uses Minimap2)

De novo Assembler (Short Reads):
• ABYSS
• SPAdes (small genomes)

Assembly Polisher:
• Apollo
• Racon
• Pilon
• Quiver (PB reads)
• Arrow (PB reads, Not published yet)
• NanoPolish (ONP reads)
Revisiting the Puzzle

“African pan-genome contains ~10% more DNA bases than the current human reference genome”
Time to Change the Reference Genome

Is it time to change the reference genome?

Sara Ballouz, Alexander Dobin & Jesse A. Gillis

*Genome Biology* 20, Article number: 159 (2019)  |  [Cite this article]

“Switching to a consensus reference would offer important advantages over the continued use of the current reference with few disadvantages”
Analysis is Bottlenecked in Read Mapping!!

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

Agenda for Today

- What is Read Mapping?
- **What Makes Read Mapper Slow?**
- Algorithmic & Hardware Acceleration
  - Seed Filtering Technique
  - Pre-alignment Filtering Technique
  - Read Alignment Acceleration
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></th>
<th>Year</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Genes &amp; Operons</td>
<td>1990</td>
<td>Kilo = 1,000</td>
</tr>
<tr>
<td>Bacterial genomes</td>
<td>1995</td>
<td>Mega = 1,000,000</td>
</tr>
<tr>
<td>Human genome</td>
<td>2000</td>
<td>Giga = 1,000,000,000</td>
</tr>
<tr>
<td>Human microbiome</td>
<td>2005</td>
<td>Tera = 1,000,000,000,000</td>
</tr>
<tr>
<td>50K Microbiomes</td>
<td>2015</td>
<td>Peta = 1,000,000,000,000,000</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th></th>
<th>Year</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>200K Microbiomes</td>
<td>2020</td>
<td>Exa = 1,000,000,000,000,000,000,000</td>
</tr>
<tr>
<td>1M Microbiomes</td>
<td>2025</td>
<td>Zetta = 1,000,000,000,000,000,000,000,000</td>
</tr>
<tr>
<td>Earth Microbiome</td>
<td>2030</td>
<td>Yotta = 1,000,000,000,000,000,000,000,000,000,000</td>
</tr>
</tbody>
</table>

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

**Specialized Machine**
for Sequencing

**General-Purpose Machine**
for Analysis

**FAST**

**SLOW**
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

Burks, Goldstein, von Neumann, “Preliminary discussion of the logical design of an electronic computing instrument,” 1946.
The Problem

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
>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
Sequence Alignment in Unavoidable Quadratic-time dynamic-programming algorithm WHY?!

Enumerating all possible prefixes

<table>
<thead>
<tr>
<th>N E T H E R L A N D S</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 10 11</td>
</tr>
</tbody>
</table>

- NETHERLANDS x SWITZERLAND
- NETHERLANDS x S
- NETHERLANDS x SW
- NETHERLANDS x SWI
- NETHERLANDS x SWIT
- NETHERLANDS x SWITZ
- NETHERLANDS x SWITZE
- NETHERLANDS x SWITZER
- NETHERLANDS x SWITZERL
- NETHERLANDS x SWITZERLA
- NETHERLANDS x SWITZERLAN
- NETHERLANDS x SWITZERLAND

WHY?!
Sequence Alignment in Unavoidable

- **Quadratic-time** dynamic-programming algorithm
  - Enumerating all possible prefixes

- **Data dependencies** limit the computation parallelism
  - Processing row (or column) after another

- **Entire matrix** is computed even though strings can be dissimilar.
  - Number of differences is computed only at the backtracking step.

<table>
<thead>
<tr>
<th></th>
<th>N</th>
<th>E</th>
<th>T</th>
<th>H</th>
<th>E</th>
<th>R</th>
<th>L</th>
<th>A</th>
<th>N</th>
<th>D</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>S</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>W</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>I</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>T</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>Z</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>E</td>
<td>6</td>
<td>6</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>R</td>
<td>7</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>L</td>
<td>8</td>
<td>8</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>A</td>
<td>9</td>
<td>9</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>N</td>
<td>10</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>D</td>
<td>11</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>
Computer Science > Computational Complexity

[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.
Large Search Space for Mapping Location

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

Xin et al, BMC genomics (2013)
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>fqcnt_rs2_needletail.rs</td>
<td>Rust</td>
<td>9.3</td>
<td>0.8</td>
<td>needletail; fasta/4-line fastq</td>
</tr>
<tr>
<td>fqcnt_c1_kseq.c</td>
<td>C</td>
<td>9.7</td>
<td>1.4</td>
<td>multi-line fasta/fastq</td>
</tr>
<tr>
<td>fqcnt_cr1_klib.cr</td>
<td>Crystal</td>
<td>9.7</td>
<td>1.5</td>
<td>kseq.h port</td>
</tr>
<tr>
<td>fqcnt_nim1_klib.nim</td>
<td>Nim</td>
<td>10.5</td>
<td>2.3</td>
<td>kseq.h port</td>
</tr>
<tr>
<td>fqcnt_jl1_klib.jl</td>
<td>Julia</td>
<td>11.2</td>
<td>2.9</td>
<td>kseq.h port</td>
</tr>
<tr>
<td>fqcnt_js1_k8.js</td>
<td>Javascript</td>
<td>17.5</td>
<td>9.4</td>
<td>kseq.h port</td>
</tr>
<tr>
<td>fqcnt_go1.go</td>
<td>Go</td>
<td>19.1</td>
<td>2.8</td>
<td>4-line only</td>
</tr>
<tr>
<td>fqcnt_lua1_klib.lua</td>
<td>LuaJIT</td>
<td>28.6</td>
<td>27.2</td>
<td>partial kseq.h port</td>
</tr>
<tr>
<td>fqcnt_py2_rfq.py</td>
<td>PyPy</td>
<td>28.9</td>
<td>14.6</td>
<td>partial kseq.h port</td>
</tr>
<tr>
<td>fqcnt_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**
Agenda for Today

- What is Read Mapping?
- What Makes Read Mapper Slow?
- **Algorithmic & Hardware Acceleration**
  - Seed Filtering Technique
  - Pre-alignment Filtering Technique
  - Read Alignment Acceleration
Accelerating Read Mapping

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.
Our Contributions

**Near-memory/In-memory Pre-alignment Filtering**

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

**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]*

---

Sequencing Machine → Storage (SSD/HDD) → Main Memory → Microprocessor

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

“Accelerating Genome Analysis: A Primer on an Ongoing Journey”
Near-memory Pre-alignment Filtering

Gagandeepr Smoke, 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]
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 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)]
Detailed Lectures on Genome Analysis

- **Computer Architecture, Fall 2020, Lecture 3a**
  - *Introduction to Genome Sequence Analysis* (ETH Zürich, Fall 2020)
  - [YouTube](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)
  - [YouTube](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)
  - [YouTube](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)
  - [YouTube](https://www.youtube.com/watch?v=rgjl8ZyLsAg&list=PL5Q2soXY2Zi9E2bBVAgCgLgwiDRQDTyId)

Visit [SAFARI](https://www.youtube.com/onurmutlulectures) for more resources.
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)


