

Accelerating Genome Analysis Using New Algorithms and Hardware Designs

Mohammed Alser

ETH Zurich

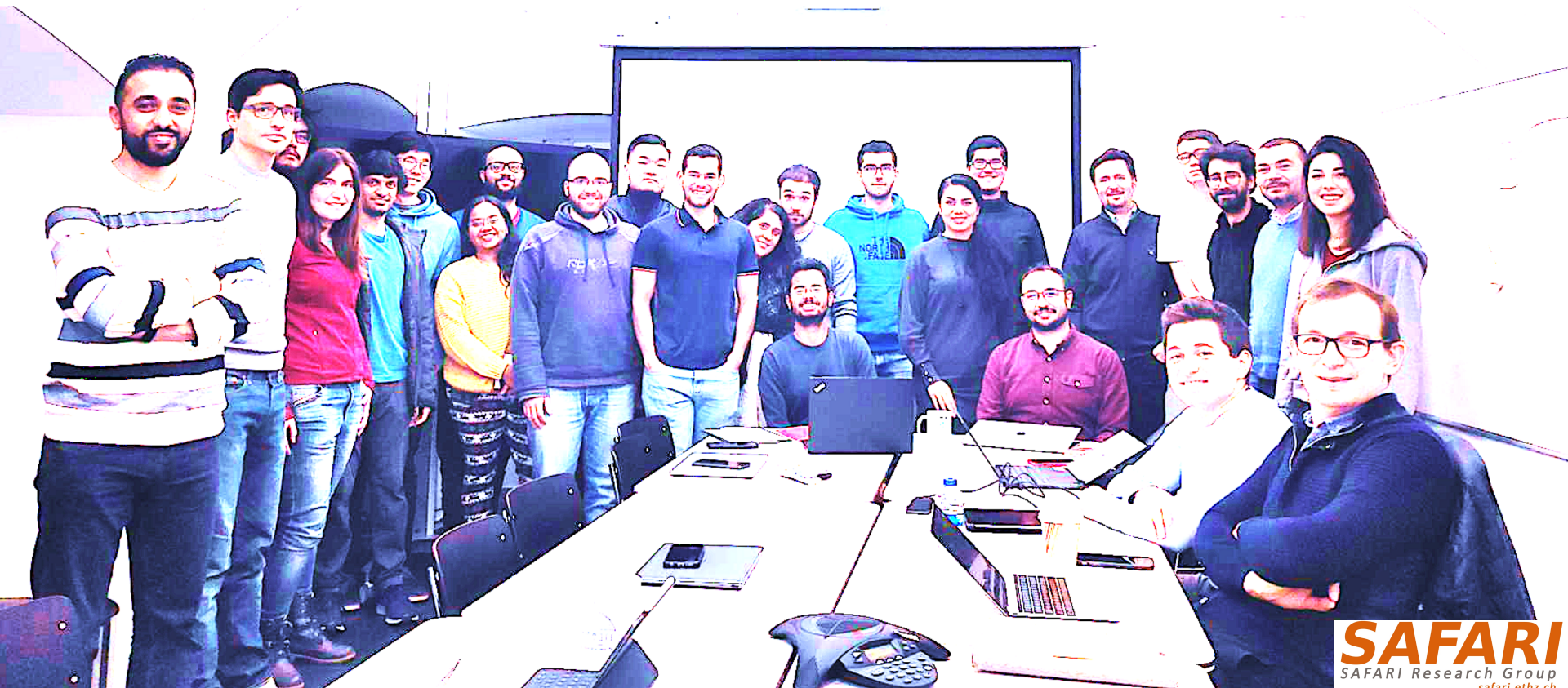
ALSERM@inf.ethz.ch

The University of Tokyo, Kashiwa Campus

18 December 2019

SAFARI Research Group

31 ☺ = 1 Professor, 2 Lecturers & Senior Researchers, 3 Senior Researchers,
12 PhD Students, 3 Masters, 8 Interns, 2 Admins



Think BIG, Aim HIGH!

Switzerland, Zurich, ETH Zurich, CS



Professor Mutlu's Bio



■ Onur Mutlu

- ❑ Professor @ ETH Zurich CS, since September'15, started May'16
- ❑ Strecker Professor @ Carnegie Mellon University ECE (CS), 2009-2016, 2016-...
- ❑ PhD from UT-Austin, worked @ Google, VMware, Microsoft Research, Intel, AMD
- ❑ <https://people.inf.ethz.ch/omutlu/>
- ❑ omutlu@gmail.com (Best way to reach me)
- ❑ Publications: <https://people.inf.ethz.ch/omutlu/projects.htm>

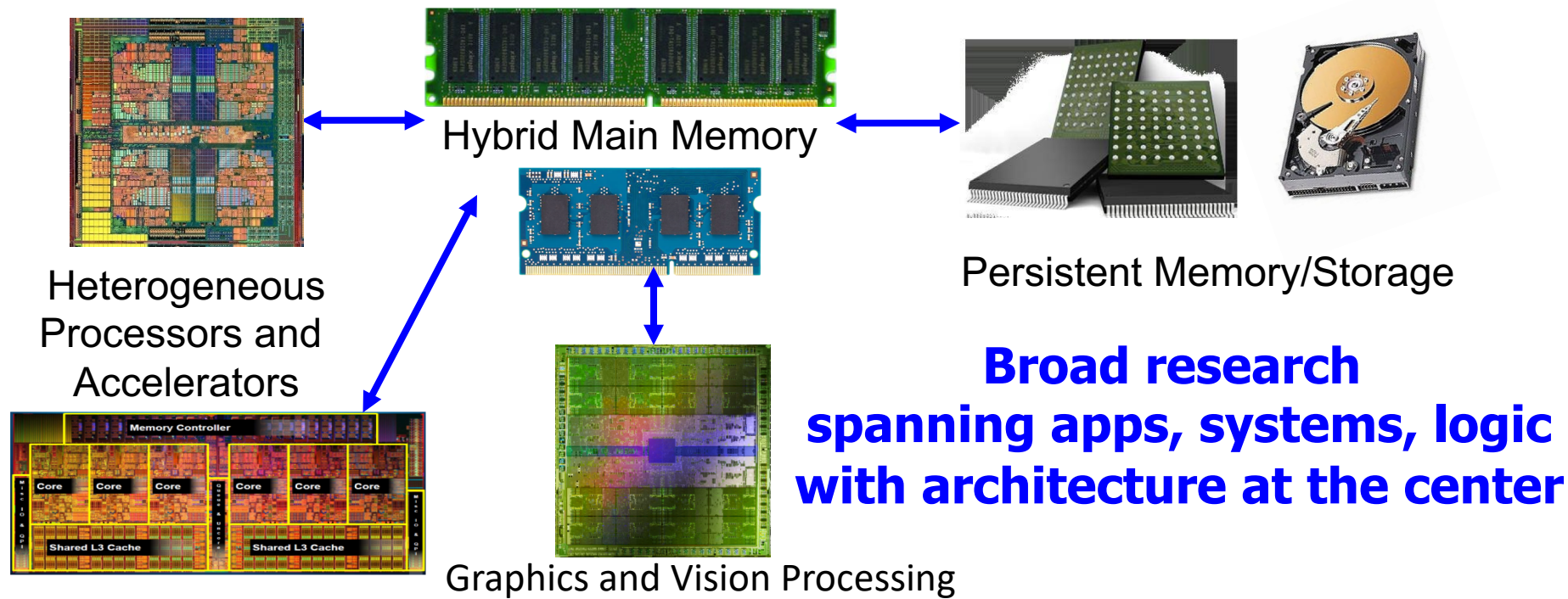
■ Research, Education, Consulting in

- ❑ Computer architecture and systems, bioinformatics
- ❑ Memory and storage systems, emerging technologies
- ❑ Many-core systems, heterogeneous systems, core design
- ❑ Interconnects
- ❑ Hardware/software interaction and co-design (PL, OS, Architecture)
- ❑ Predictable and QoS-aware systems
- ❑ Hardware fault tolerance and security
- ❑ Algorithms and architectures for genome analysis
- ❑ ...

Current Research Focus Areas

Research Focus: *Computer architecture, HW/SW, security, bioinformatics*

- **Memory and storage** (DRAM, flash, emerging), interconnects, security
- **Heterogeneous & parallel systems**, GPUs, systems for data analytics
- **System/architecture interaction**, new execution models, new interfaces
- **Energy efficiency**, fault tolerance, hardware security, performance
- **Genome sequence analysis** & assembly algorithms and architectures
- **Biologically inspired systems** & system design for bio/medicine



Openings @ SAFARI

- We are **hiring** enthusiastic and motivated students and researchers at all levels.
- Join us now:

safari.ethz.ch/apply

We ♥ Japan



SAFARI

Agenda for Today

- This lecture is **NOT** about how to **analyze biological** data using available tools.

Agenda for Today

- Why Genome Analysis?
- What is Genome Analysis?
- How we Map Reads?
- What Makes Read Mapper Slow?
- Algorithmic & Hardware Acceleration
 - Seed Filtering Technique
 - Pre-alignment Filtering Technique
 - Read Alignment Acceleration
- Where is Read Mapping Going Next?

Agenda for Today

- Why Genome Analysis?
- What is Genome Analysis?
- How we Map Reads?
- What Makes Read Mapper Slow?
- Algorithmic & Hardware Acceleration
 - Seed Filtering Technique
 - Pre-alignment Filtering Technique
 - Read Alignment Acceleration
- Where is Read Mapping Going Next?

Why Genome Analysis? Why Bother?

- Personalized medicine.
- Genome-wide association study (GWAS).
- City-scale microbiome profiling.
- Tracing birth parents.
- Disease risk profiling.
- ...

1-Personalized Medicine

Nan-Byo

Difficult + Illness

Coined in 1972 by the Japanese Ministry of Labor, Health, and Welfare.

<https://www.nanbyo-research.jp/nanbyo>

難病

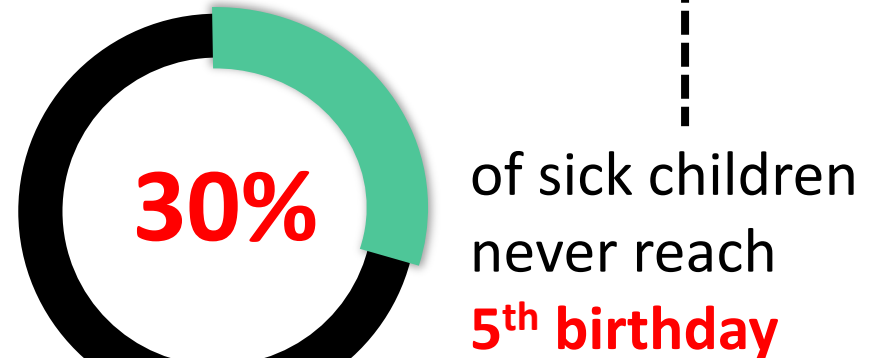
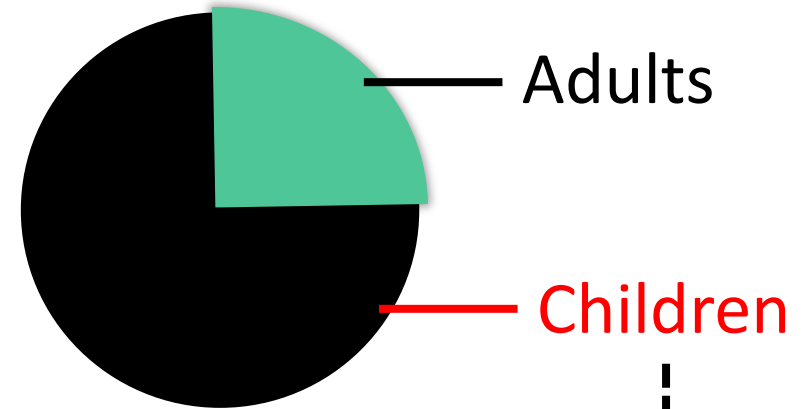
1-Personalized Medicine



1 in 17 people

in the world have a rare disease

That's 350 Million



Rare Diseases in Japan

“**We don’t know exactly** how many people in Japan have a rare disease, which is why we want to design the rare disease platform to be as comprehensive as possible. There are thousands of rare diseases. So even though the number of patients with each disease is very small, there are many people who have one. **Out of 20 of your friends, for example, one will have a rare disease,**” explains Matsuda.



Prof. Fumihiko Matsuda,
Director of the Center for
Genomic Medicine,
Kyoto University




<https://www.nanbyo-research.jp/feature/43/japan%E2%80%99s-rare-disease-database-expedites-more-effective-research>

Personalized Medicine in Japan

European Journal of
Human Genetics

Policy | [Open Access](#) | Published: 05 July 2017

Japan's initiative on rare and undiagnosed diseases (IRUD): towards an end to the diagnostic odyssey

Takeya Adachi , Kazuo Kawamura, Yoshihiko Furusawa, Yuji Nishizaki, Noriaki Imanishi, Senkei Umehara , Kazuo Izumi  & Makoto Suematsu

European Journal of Human Genetics **25**, 1025–1028(2017) | [Cite this article](#)



> 2000
undiagnosed
patients



> 600 million
JPY annually

Personalized Medicine in UK

npj | Genomic Medicine

www.nature.com/npjgenmed

PMCID: PMC5884823

PMID: [29644095](https://pubmed.ncbi.nlm.nih.gov/29644095/)

[NPJ Genom Med](#). 2018; 3: 10.

Published online 2018 Apr 4. doi: [10.1038/s41525-018-0049-4](https://doi.org/10.1038/s41525-018-0049-4)

Rapid whole-genome sequencing decreases infant morbidity and cost of hospitalization

[Lauge Farnaes](#),^{#1,2} [Amber Hildreth](#),^{#1,2} [Nathaly M. Sweeney](#),^{#1,2} [Michelle M. Clark](#),¹ [Shimul Chowdhury](#),¹ [Shareef Nahas](#),¹ [Julie A. Cakici](#),¹ [Wendy Benson](#),¹ [Robert H. Kaplan](#),³ [Richard Kronick](#),⁴ [Matthew N. Bainbridge](#),¹ [Jennifer Friedman](#),^{1,2,5} [Jeffrey J. Gold](#),^{1,5} [Yan Ding](#),¹ [Narayanan Veeraraghavan](#),¹ [David Dimmock](#),¹ and [Stephen F. Kingsmore](#)^{✉1}



reduced inpatient
cost by

**\$100,000-
\$300,000**

NHS
*National Institute for
Health Research*

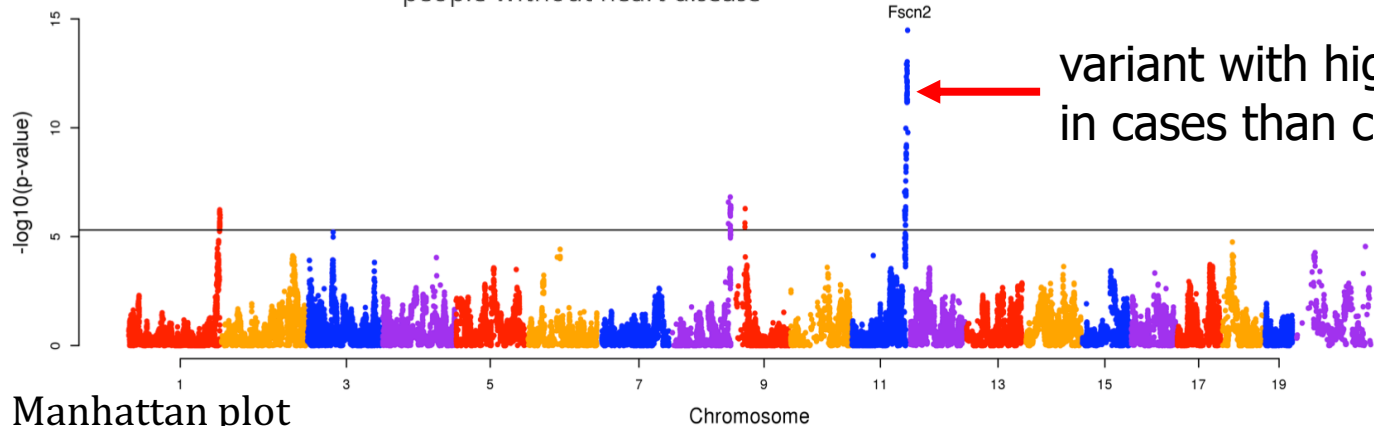
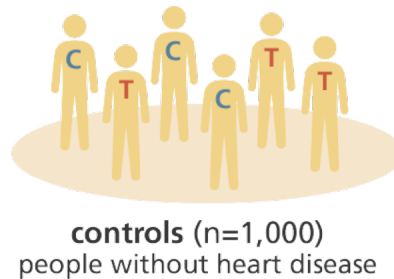
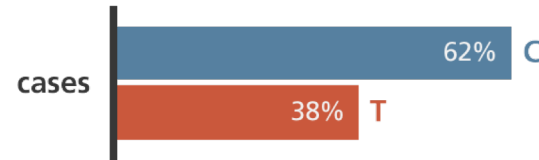
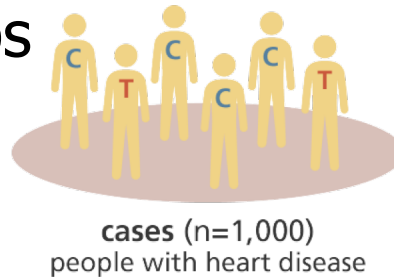
“From 2019, **all seriously ill children**
in UK will be offered **whole genome
sequencing** as part of their care”

SAFARI

Farnaes+, “[Rapid whole-genome sequencing decreases infant morbidity and cost of hospitalization](#)”, NPJ Genom Med. 2018


2-Genome-Wide Association Study (GWAS)

- Detecting genetic variants associated with phenotypes using two groups of people.



Finding SNPs Associated with Complex Trait

	SNP1	SNP2	Blood Pressure
Different individuals	...ACATG C CGACATTTCATAG G GCC...		180
	...ACATG C CGACATTTCATAG A GCC...		175
	...ACATG C CGACATTTCATAG G GCC...		170
	...ACATG C CGACATTTCATAG A GCC...		165
	...ACATG C CGACATTTCATAG G GCC...		160
	...ACATG C CGACATTTCATAG G GCC...		145
	...ACATG C CGACATTTCATAG A GCC...		140
	...ACATG C CGACATTTCATAG A GCC...		130
	...ACATG T CGACATTTCATAG G GCC...		120
	...ACATG T CGACATTTCATAG A GCC...		120
	...ACATG T CGACATTTCATAG G GCC...		115
	...ACATG T CGACATTTCATAG A GCC...		110
	...ACATG T CGACATTTCATAG G GCC...		110
	...ACATG T CGACATTTCATAG A GCC...		110
	...ACATG T CGACATTTCATAG G GCC...		105
	...ACATG T CGACATTTCATAG A GCC...		100



Eleazar Eskin: Discovering the Causal Variants Involved in GWAS Studies, CGSI 2018, UCLA

Mirror Phenotypes of 593 Kb CNVs



AUTISM

Weiss, *N Eng J Med* 2008
Deletion of 593 kb



SCHIZOPHRENIA

McCarthy, *Nat Genet* 2009
Duplication of 593 kb



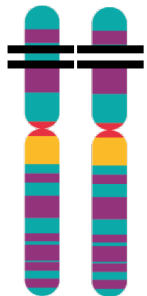
OBESITY

Walters, *Nature* 2010
Deletion of 593 kb



UNDERWEIGHT

Jacquemont, *Nature* 2011
Duplication of 593 kb



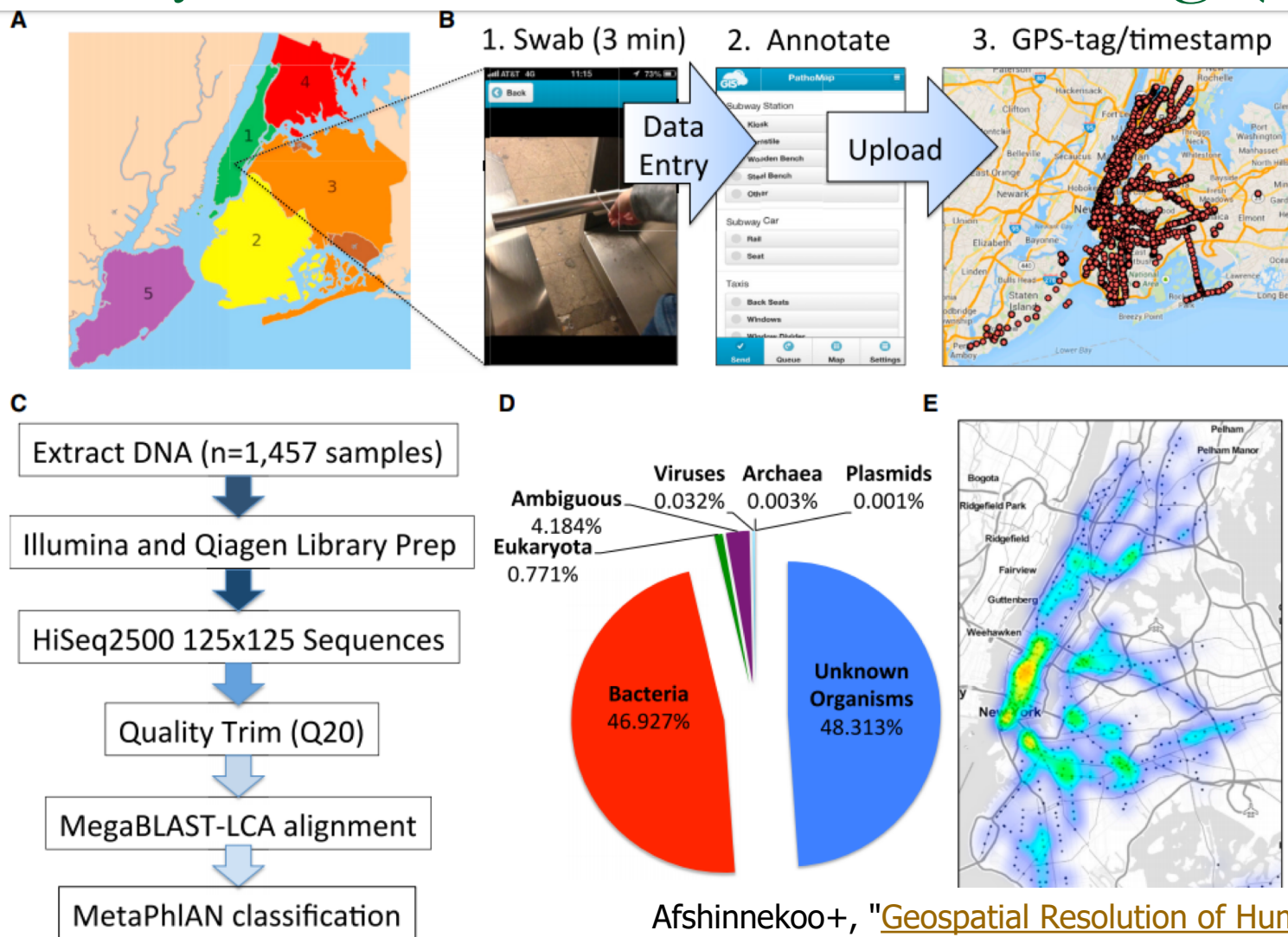
Deletion in the short arm
of chromosome 16 (16p11.2)



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

[illegible]

3- City-Scale Microbiome Profiling (cont'd)



Afshinnekoo+, "Geospatial Resolution of Human and Bacterial Diversity with City-Scale Metagenomics", Cell Systems, 2015

Figure 1. The Metagenome of New York City

(A) The five boroughs of NYC include (1) Manhattan (green), (2) Brooklyn (yellow), (3) Queens (orange), (4) Bronx (red), (5) Staten Island (lavender).

(B) The collection from the 466 subway stations of NYC across the 24 subway lines involved three main steps: (1) collection with Copan Elution swabs, (2) data entry into the database, and (3) uploading of the data. An image is shown of the current collection database, taken from <http://pathomap.giscloud.com>.

(C) Workflow for sample DNA extraction, library preparation, sequencing, quality trimming of the FASTQ files, and alignment with MegaBLAST and MetaPhlAn to

Plague in New York Subway System?

Plague (Yersinia Pestis)



Harvard Health Publishing
HARVARD MEDICAL SCHOOL

Trusted advice for a healthier life

What Is It?

Published: December, 2018

Plague is caused by *Yersinia pestis* bacteria. It can be a life-threatening infection if not treated promptly. Plague has caused several major epidemics in Europe and Asia over the last 2,000 years. Plague has most famously been called "the Black Death" because it can cause skin sores that form black scabs. A plague epidemic in the 14th century killed more than one-third of the population of Europe within a few years. In some cities, up to 75% of the population died within days, with fever and swollen skin sores.

Plague in New York Subway System?

Plague (Yersinia)

What Is It?

Published: December, 2018

Plague is caused by *Yersinia* treated promptly. Plague has last 2,000 years. Plague has cause skin sores that form b than one-third of the popul the population died within

The New York Times
Bubonic Plague in the Subway System? Don't Worry About It



In October, riders were not deterred after reports that an Ebola-infected man had ridden the subway just before he fell ill. Robert Stolarik for The New York Times

<https://www.nytimes.com/2015/02/07/nyregion/bubonic-plague-in-the-subway-system-dont-worry-about-it.html>

The findings of *Yersinia Pestis* in the subway received wide coverage in the lay press, causing some alarm among New York residents

Failure of Bioinformatics



data. Rob Knight, a professor in the department of pediatrics at the University of California, San Diego, calls this type of error “a **failure of bioinformatics**,” in that Mason had assumed the gene fragments were unique to the pathogens, when in fact they can also be detected in other

Charles Schmidt, “[Living in a microbial world](https://www.nature.com/articles/nbt.3868)”, *Nature Biotechnology*, 2017

<https://www.nature.com/articles/nbt.3868>

There is a critical need for **fast** and
accurate genome analysis.

Agenda for Today

- Why Genome Analysis?
- What is Genome Analysis?
- How we Map Reads?
- What Makes Read Mapper Slow?
- Algorithmic & Hardware Acceleration
 - Seed Filtering Technique
 - Pre-alignment Filtering Technique
 - Read Alignment Acceleration
- Where is Read Mapping Going Next?

Genome Analysis



NO machine can read the *entire* content of a genome

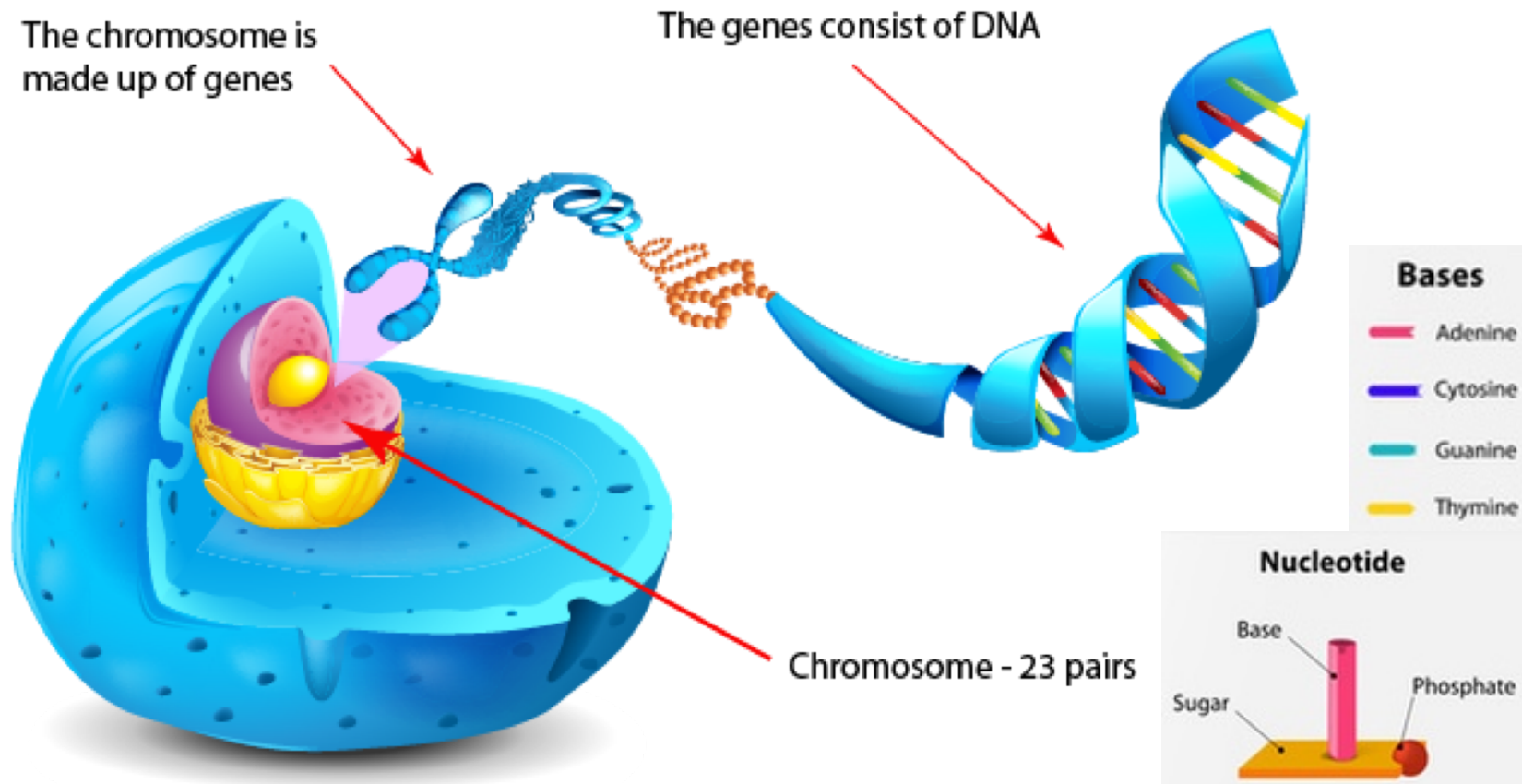


```
>CCTCCTCAGTGCCACCCAGCCCACTGGCAGCTCCCAAACAGGCTCTTATTTAAACACCCTGTTCCCTGCCCTTGGAGTGAGGTG
TCAAGGACCTAACTAAAAAAAAAAAAAAAAAGAAAAAGAAAAAGAAAAAGAAATTTAAATTTAAGTAATTCTTTGAAAAAACTAATTTCT
AAGCTTCTTCATGTCAAGGACCTAATGTGCTAAACAGCACTTTTTTGACCATTATTTGGATCTGAAAGAAATCAAGAATAAATGAA
GGACTTGATACATTGGAAGAGGAGAGTCAAGGACCTACAGAAAAAAAAAAAAAAAAAGAAAAAGAAAAAGAAAAAGAAATTTAAATTTAA
GTAATTCTTTGAAAAAACTAATTTCTAAGCTTCTTCATGTCAAGGACCTAATGTCTGTGTTGCAGGTCTTCTTGCATTTCCCTGTC
AAAAGAAAAAGAAATTTAAATTTAAGTAATTCTTTGAAAAAACTAATTTCTAAGCTTCTTCATGTCAAGGACCTAATGTCAGGCCA
AGAGTTGCAAAAAAAAAAAAAAAAAAGAAAAAGAAAAAGAAAAAGAAATTTAAATTTAAGTAATTCTTTGAAAAAACTAATTTCTAAGCTTC
TTCATGTCAAGGACCTAATGTAGCCAGAATGGTTGTGGGATGGGAGCCTCTGTGGACCGACCAGGTAGCTCTCTTTCCACACTGT
AGTCTCAAAGCTTCTTCATGTGGTTTCTCTGAGTGAAAAAAAAAAAAAAAAAGAAAAAGAAAAAGAAAAAGAAATTTAAATTTAAGTAATTC
TTTGAAAAAACTAATTTCTAAGCTTTTCATGTCAAGGACCTAATGTAGCTATACTGAACGTTATCTAGGGGAAAGATTGAAGGG
GAGCTCTAAGGTCAACACACCACCACTTCCCAGAAAGCTTCTTCATCCGTTTCTCTCCACA
```

.....

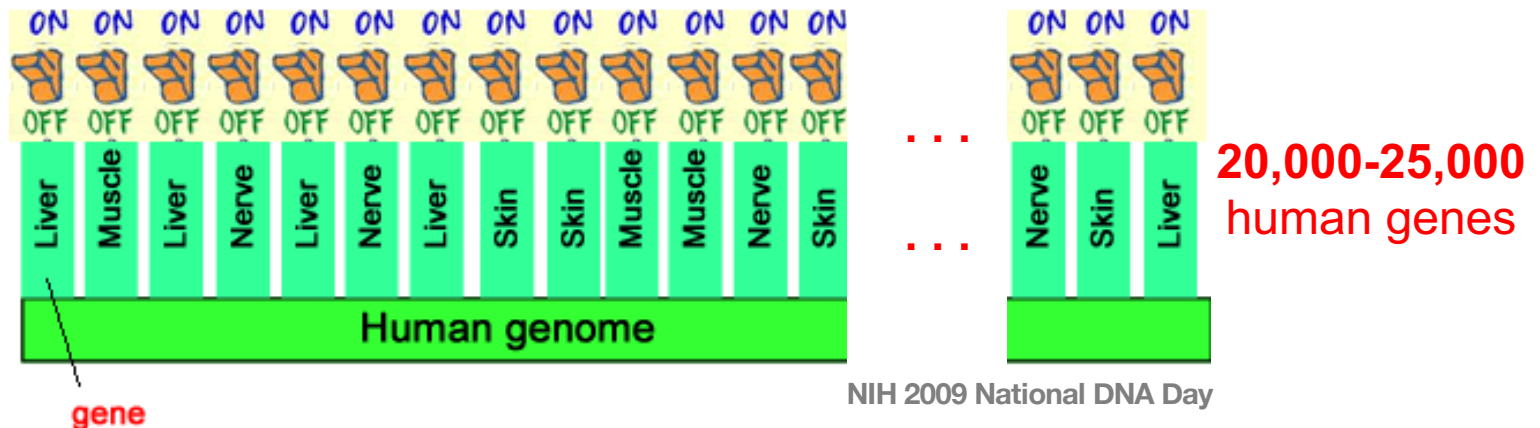
Life Begins with Cell

- A cell is a **smallest** structural unit of an organism that is capable of **independent functioning**.
 - Cells store *all* information to **replicate** themselves.

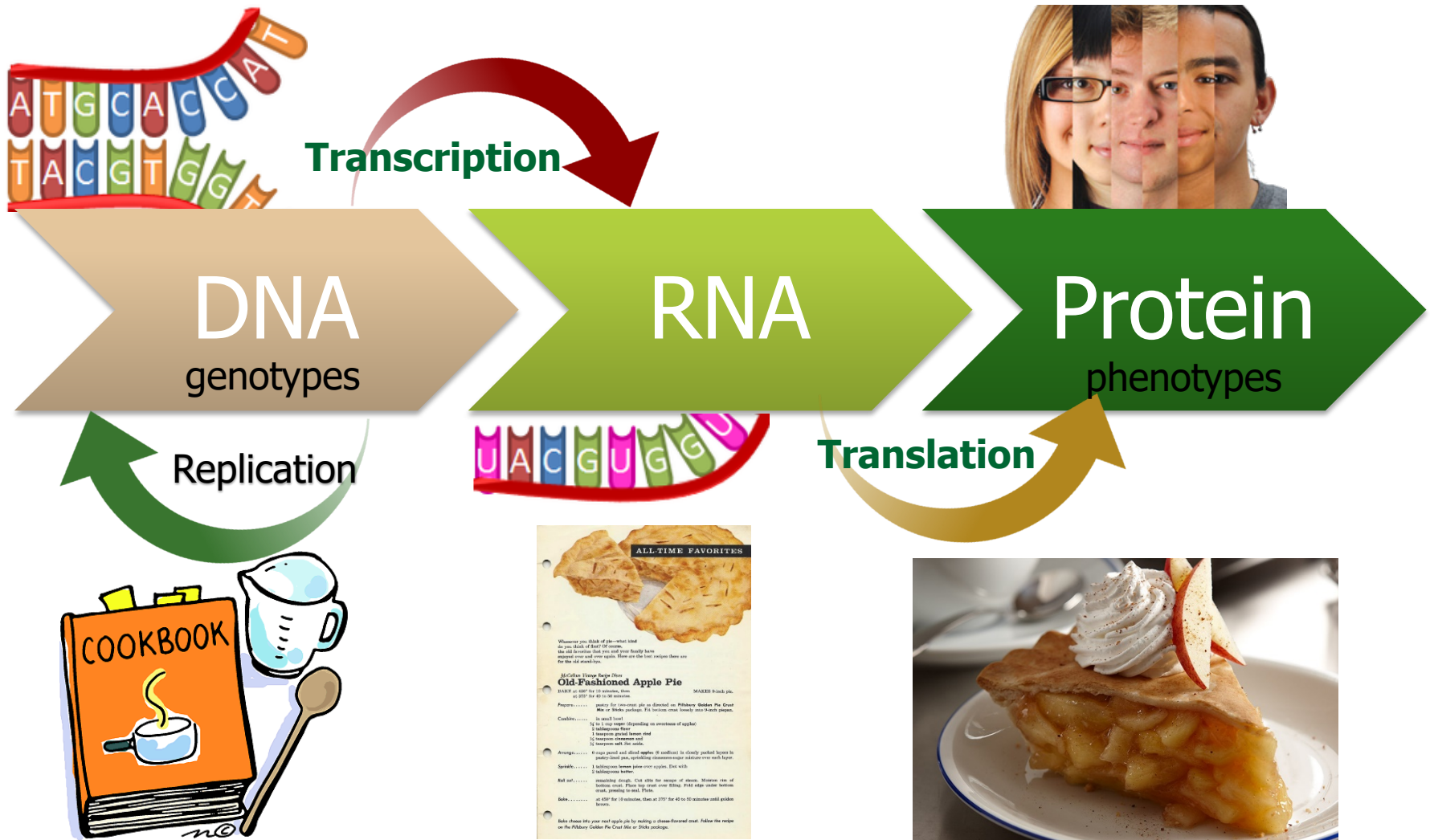


Cells of Different Organs and Tissues

- All the **cells** in a person's body have the **same DNA** and the **same genes**.
 - **Expression** of the genes **differs** between cells.
 - But **not all genes** are used or expressed by those cells.

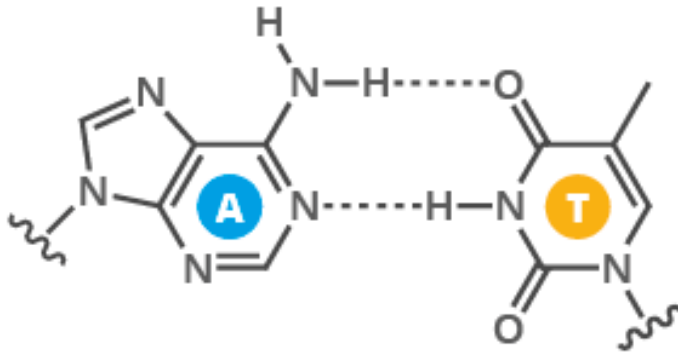


All Life Depends on 3 Critical Molecules

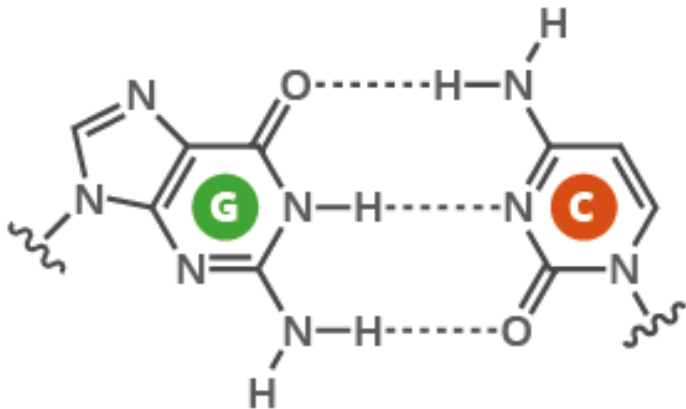


Chemical Structure of DNA

A ADENINE **T** THYMINE

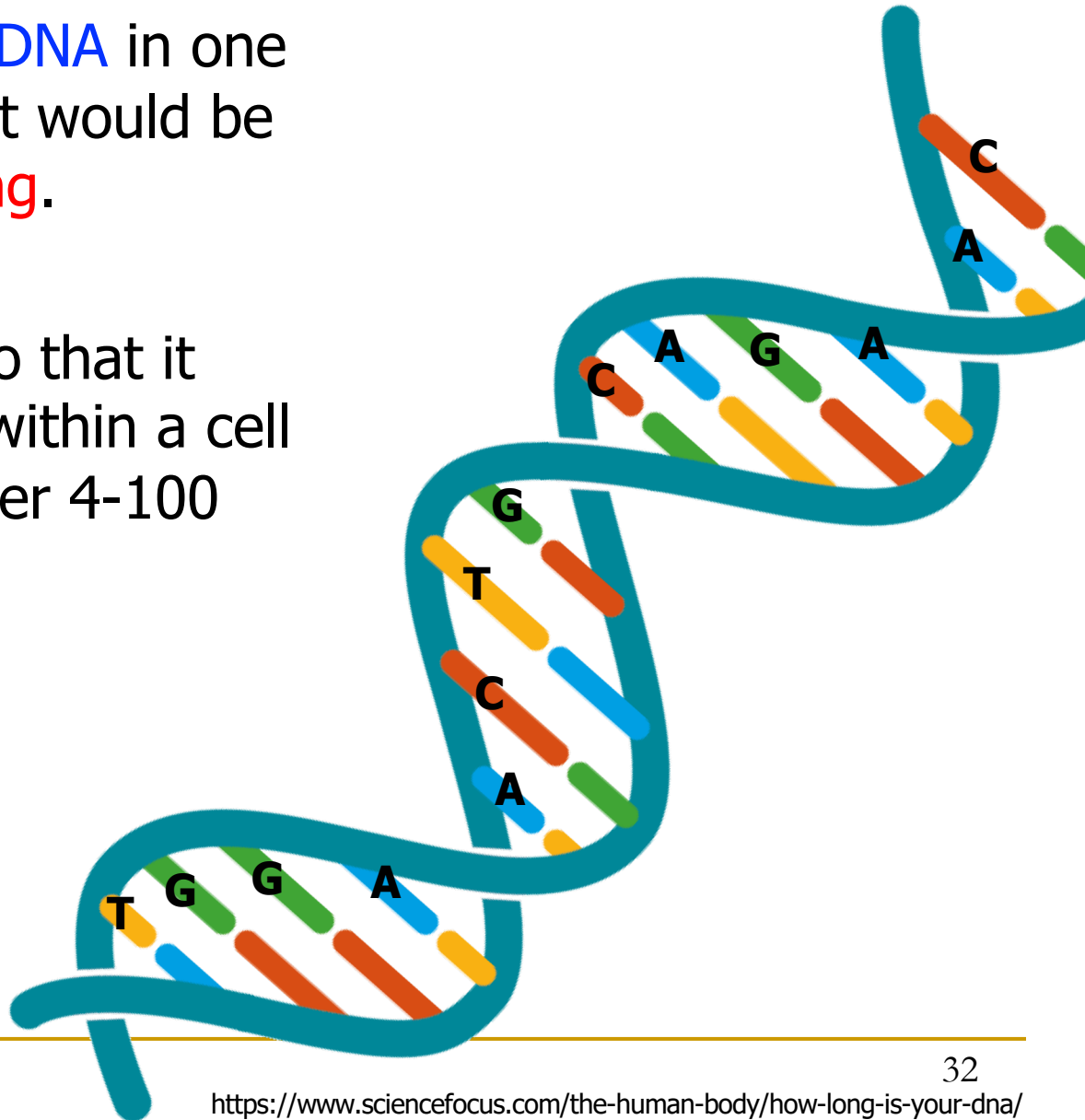


G GUANINE **C** CYTOSINE

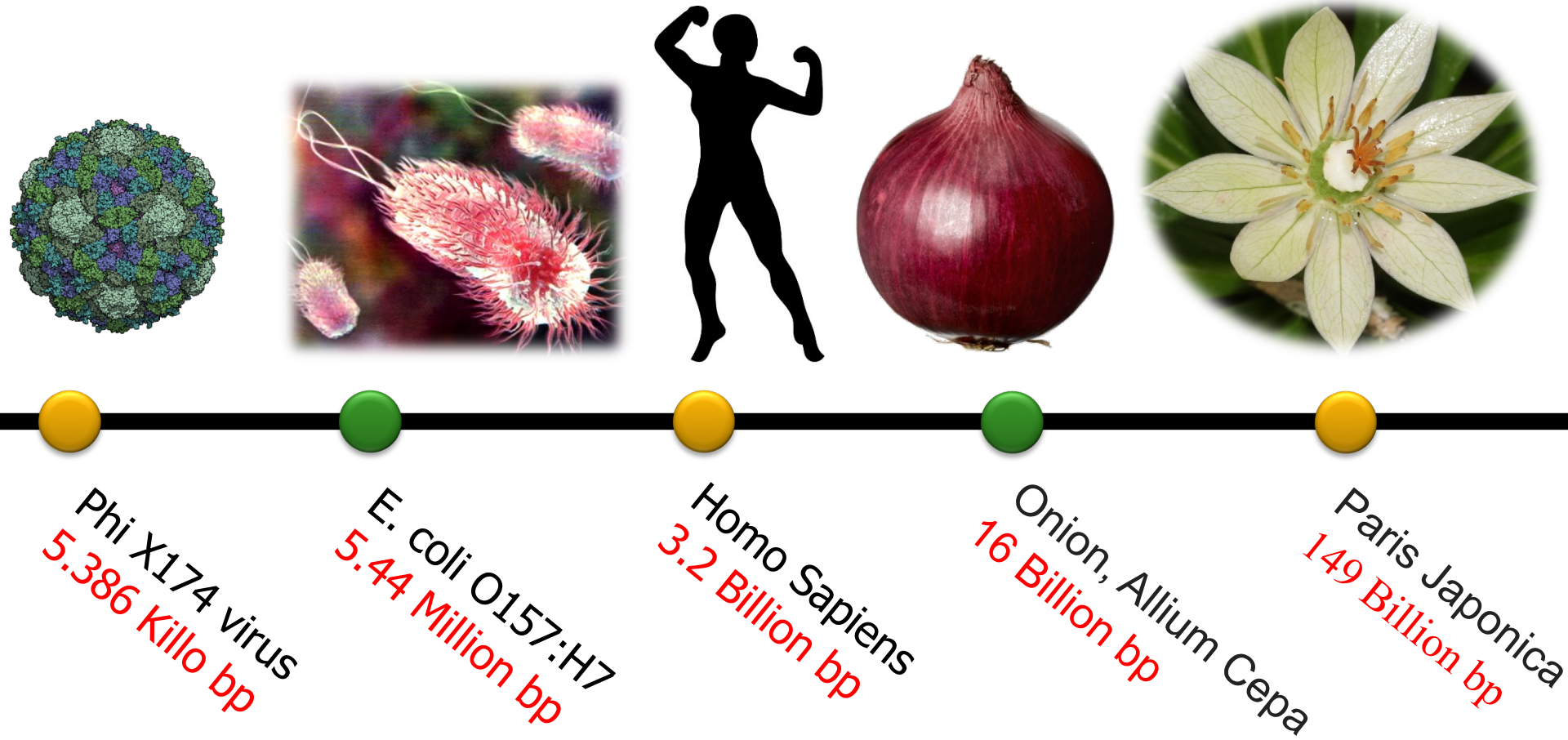


Chemical Structure of DNA

- If you stretched the DNA in one cell all the way out, it would be about 2-3 meters long.
- DNA is supercoiled so that it takes up less space within a cell (human cell's diameter 4-100 microns).



How Long is DNA?



The Genetic Similarity Between Species



Human ~ Human
99.9%



Human ~ Chimpanzee
96%



Human ~ Cat
90%

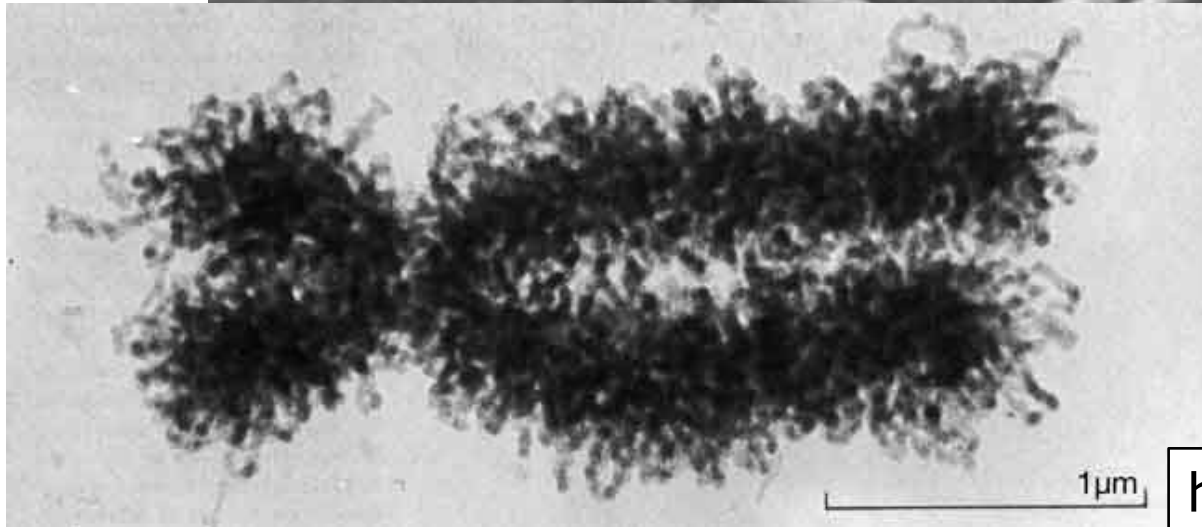
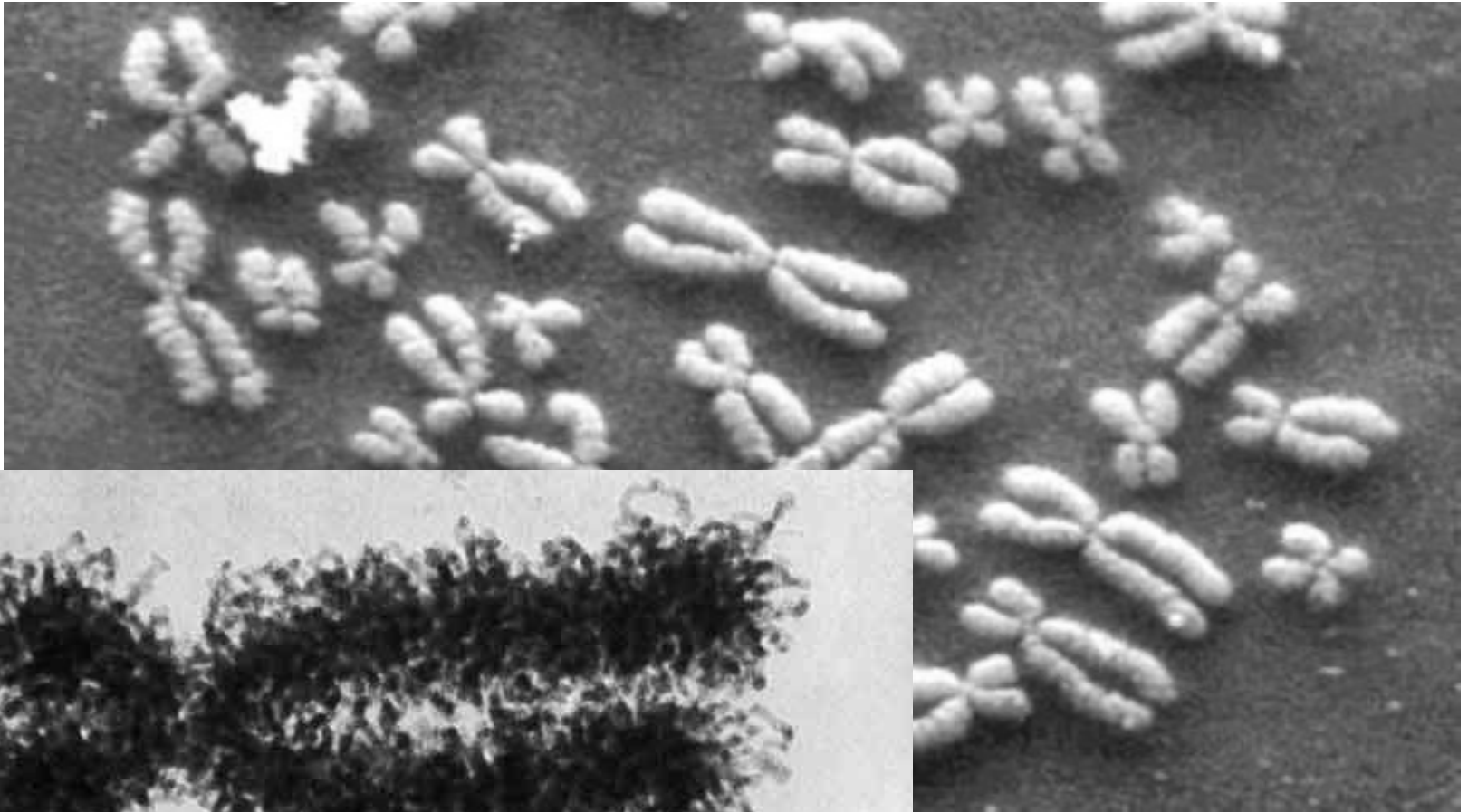


Human ~ Cow
80%



Human ~ Banana
50-60%

DNA Under Electron Microscope



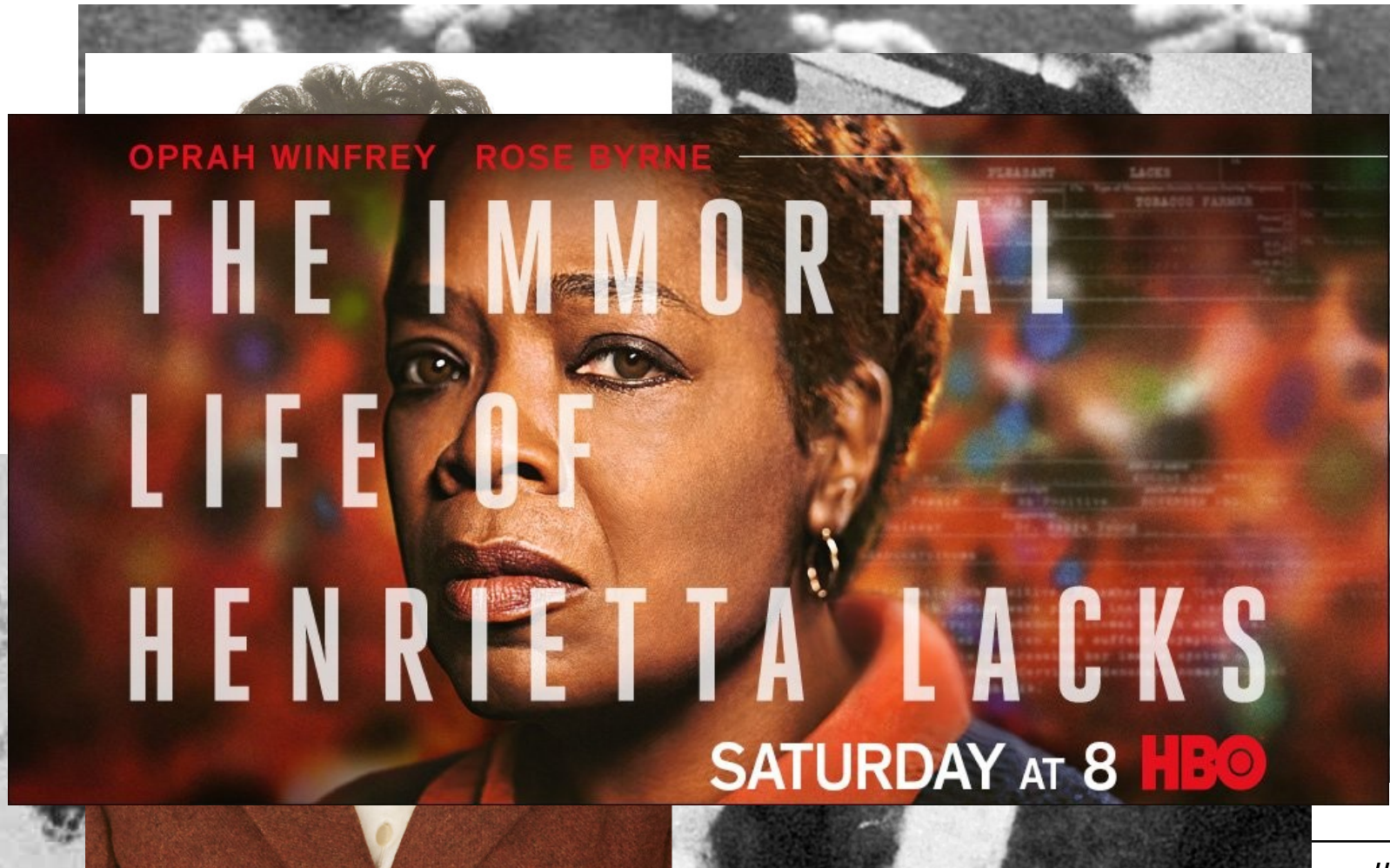
human chromosome #12
from HeLa's cell

DNA Under Electron Microscope



human chromosome #12
from HeLa's cell

DNA Under Electron Microscope



human chromosome #12
from HeLa's cell

Untangling Yarn Balls & DNA Sequencing





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.



A C 3.2×10^9
G T bases

 13 years

 $> 3 \times 10^9$ \$

Vast Improvement in Sequencing



CCCCCTATATATACGTACTAGTACGT
ACGACTTTAGTACGTACGT
TATATATACGTACTAGTACGT
ACGTACGCCCCTACGTA
TATATATACGTACTAGTACGT
ACGACTTTAGTACGTACGT
TATATATACGTACTAAAGTACGT
TATATATACGTACTAGTACGT
ACGTTTTTAAACGTA
TATATATACGTACTAGTACGT
ACGACGGGGAGTACGTACGT



1×10^{12} bases^{*}



44 hours^{*}



<1000 \$

^{*} NovaSeq 6000

High-Throughput Sequencers



Illumina MiSeq



Pacific
Biosciences
Sequel II

Oxford
Nanopore
PromethION



Illumina NovaSeq 6000



Pacific Biosciences RS II



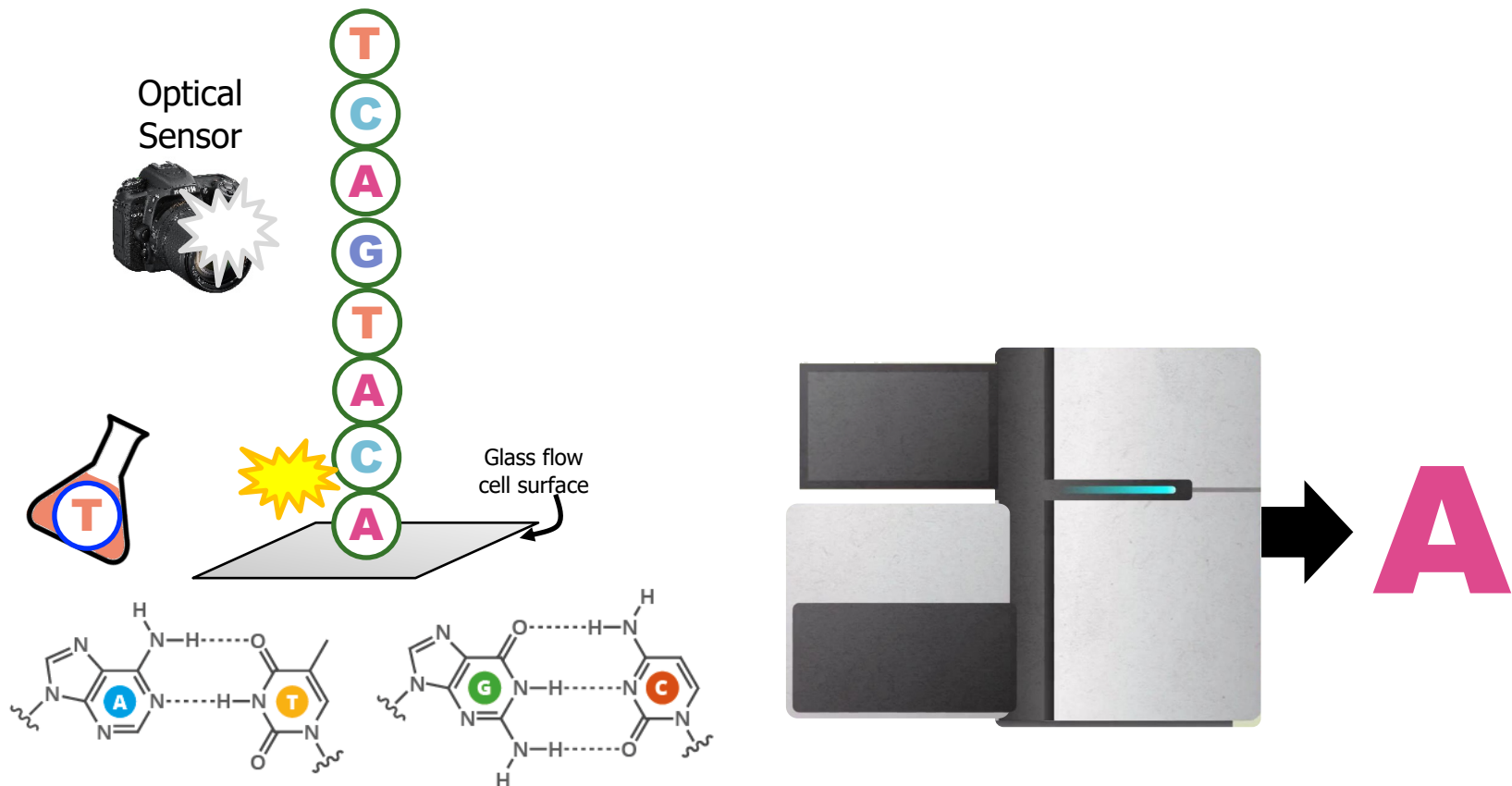
Oxford Nanopore MinION



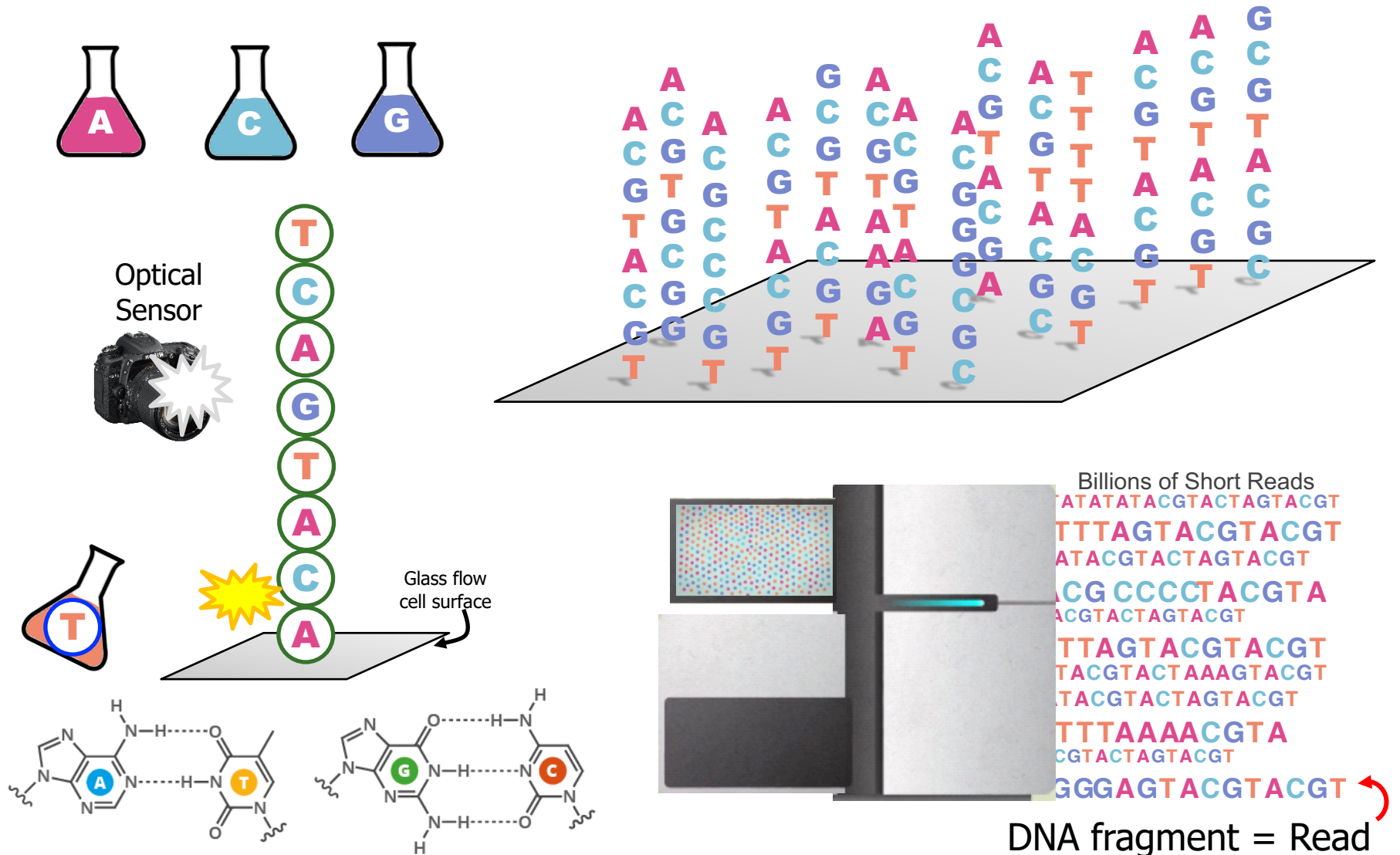
Oxford
Nanopore
SmidgION

... and more! All produce data with different properties.

How Does HTS Machine Work?

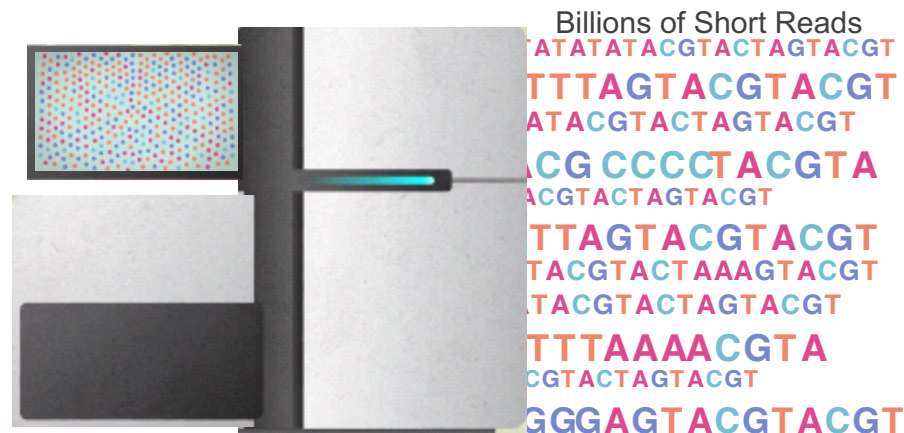


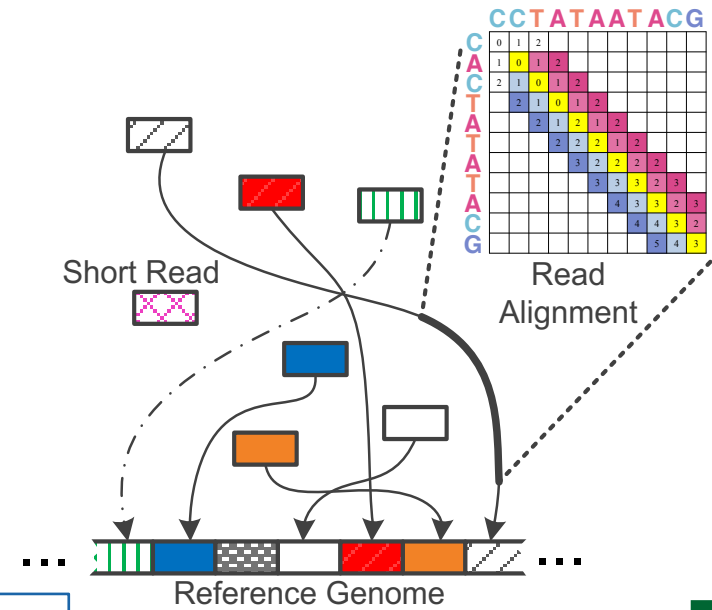
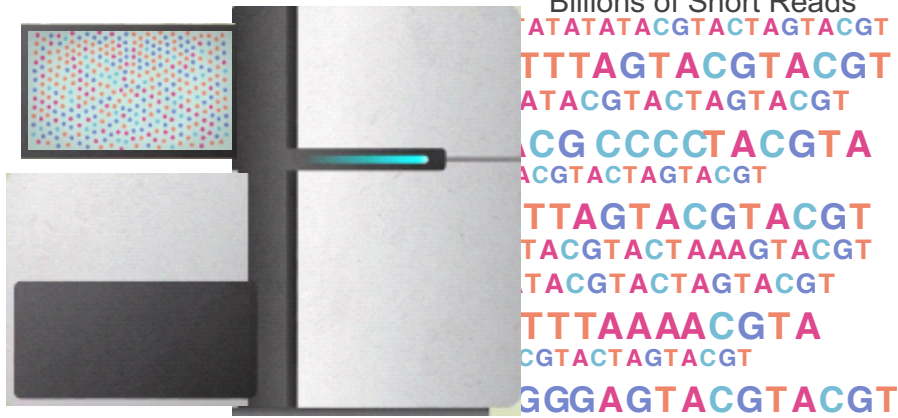
How Does HTS Machine Work?



How Does HTS Machine Work?

Reads lack information about their order and location (which part of genome they are originated from)





1 Sequencing

Genome Analysis

2 Read Mapping

reference: TTTATCGCTTCCATGACGCAG
 read1: ATCGCATCC
 read2: TATCGCATC
 read3: CATCCATGA
 read4: CGCTTCCAT
 read5: CCATGACGC
 read6: TTCCATGAC



3 Variant Calling

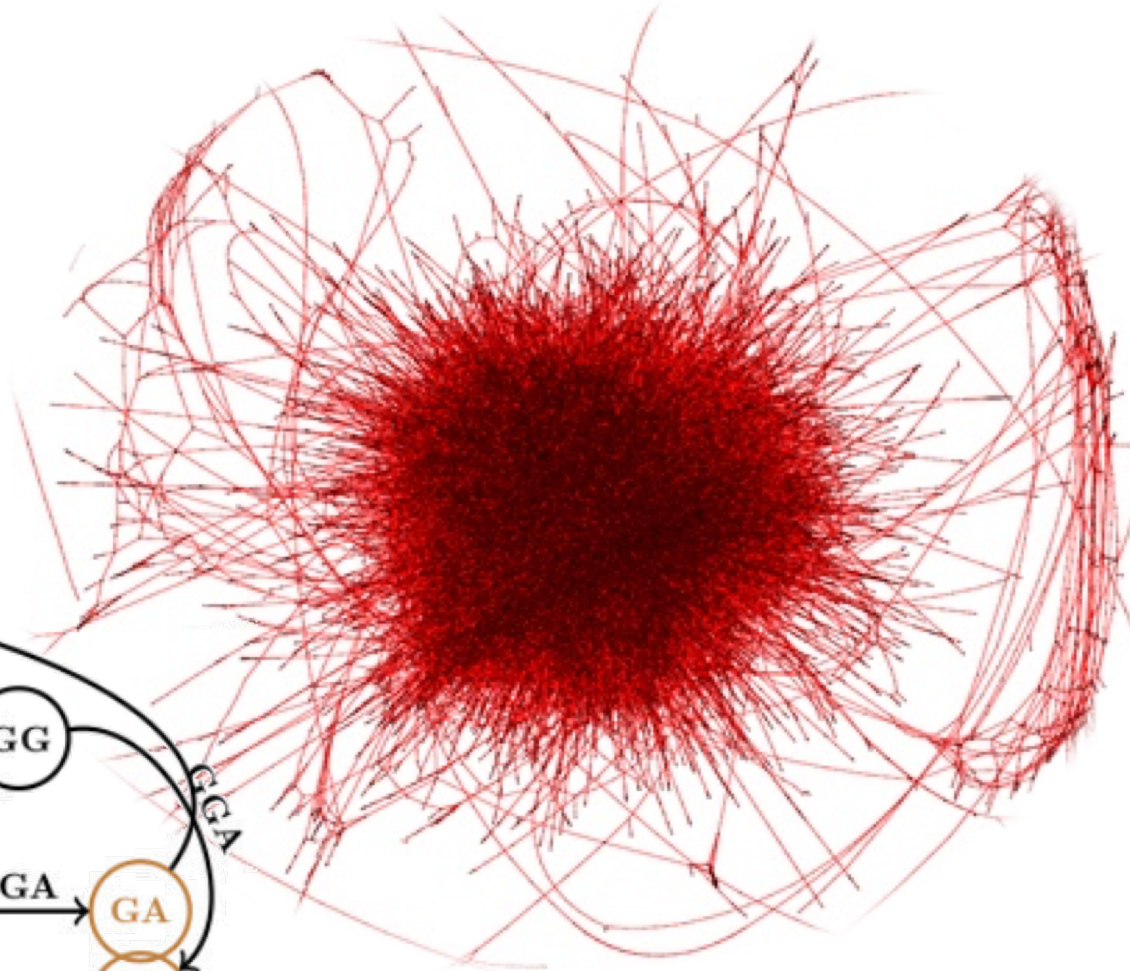
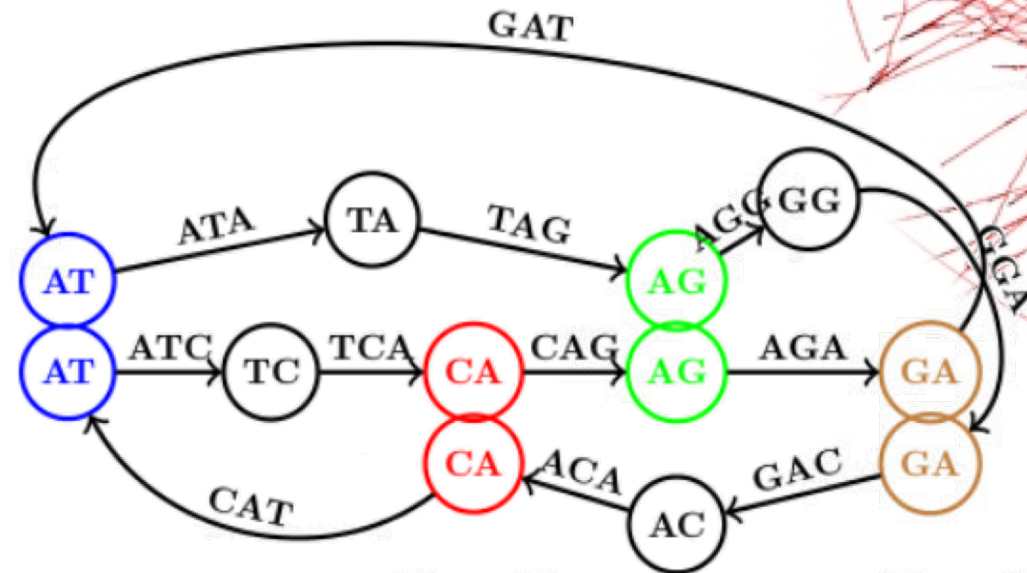
4 Scientific Discovery

Building up the Donor's Genome



De Novo Genome Assembly

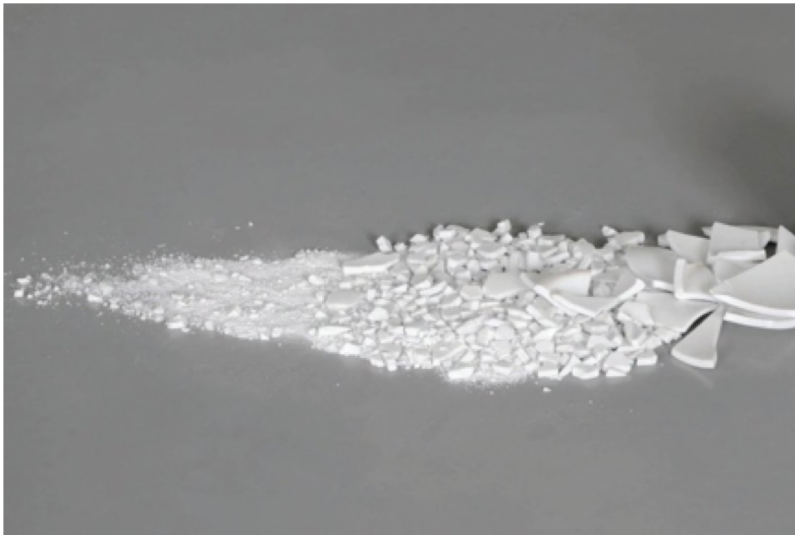
Reference-free



computationalgenomics.bioinformatics.ucla.edu/portfolio/david-koslicki-the-cami-project-assessment-of-computational-techniques-in-metagenomics/

HTS Sequencing Output

Small pieces of a broken vase
short reads



Large pieces of a broken vase
long reads



Which sequencing technology is the best?

☐ 50-300 bp

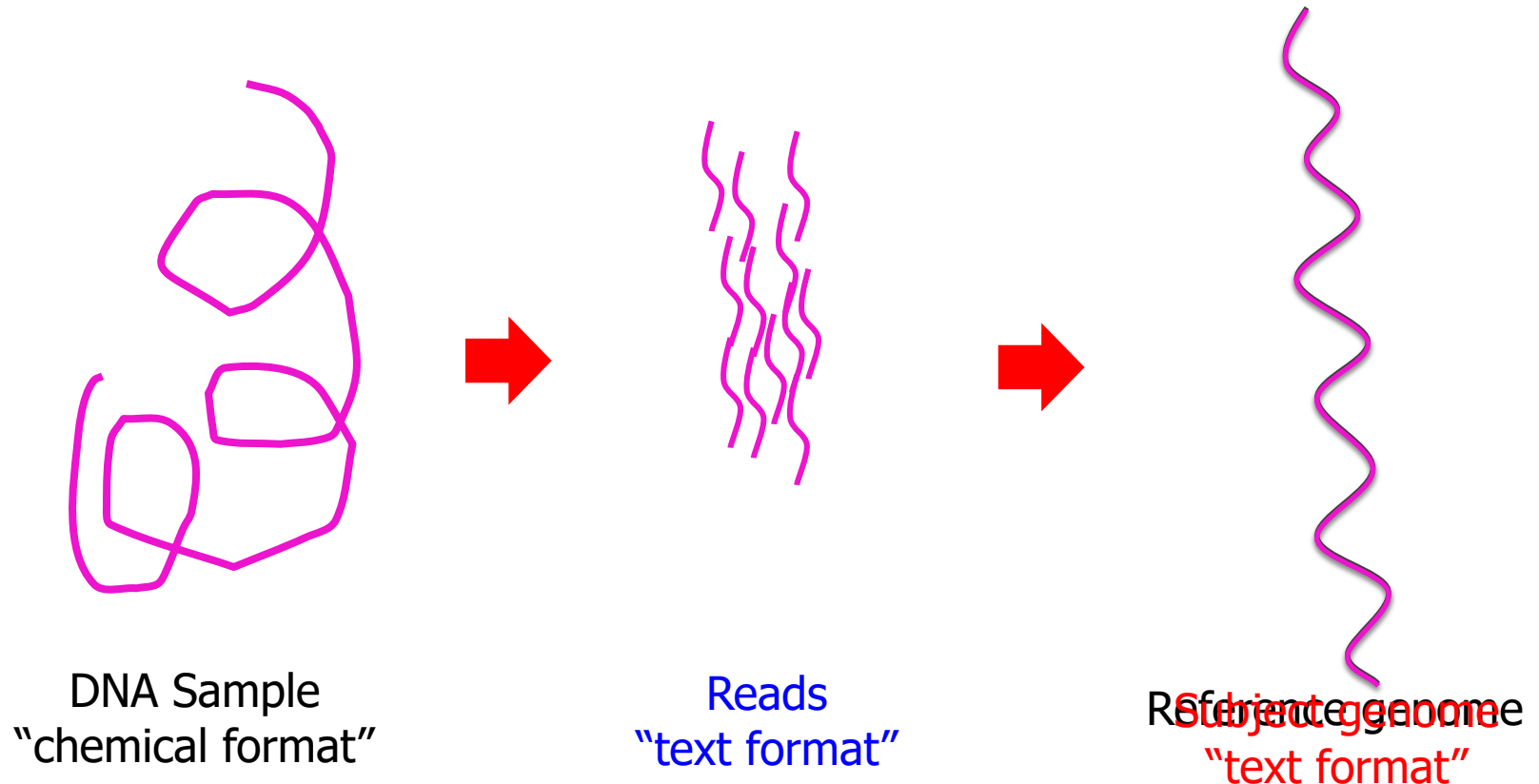
☐ low error rate ($\sim 0.1\%$)

☐ 10K-100K bp

☐ high error rate ($\sim 15\%$)

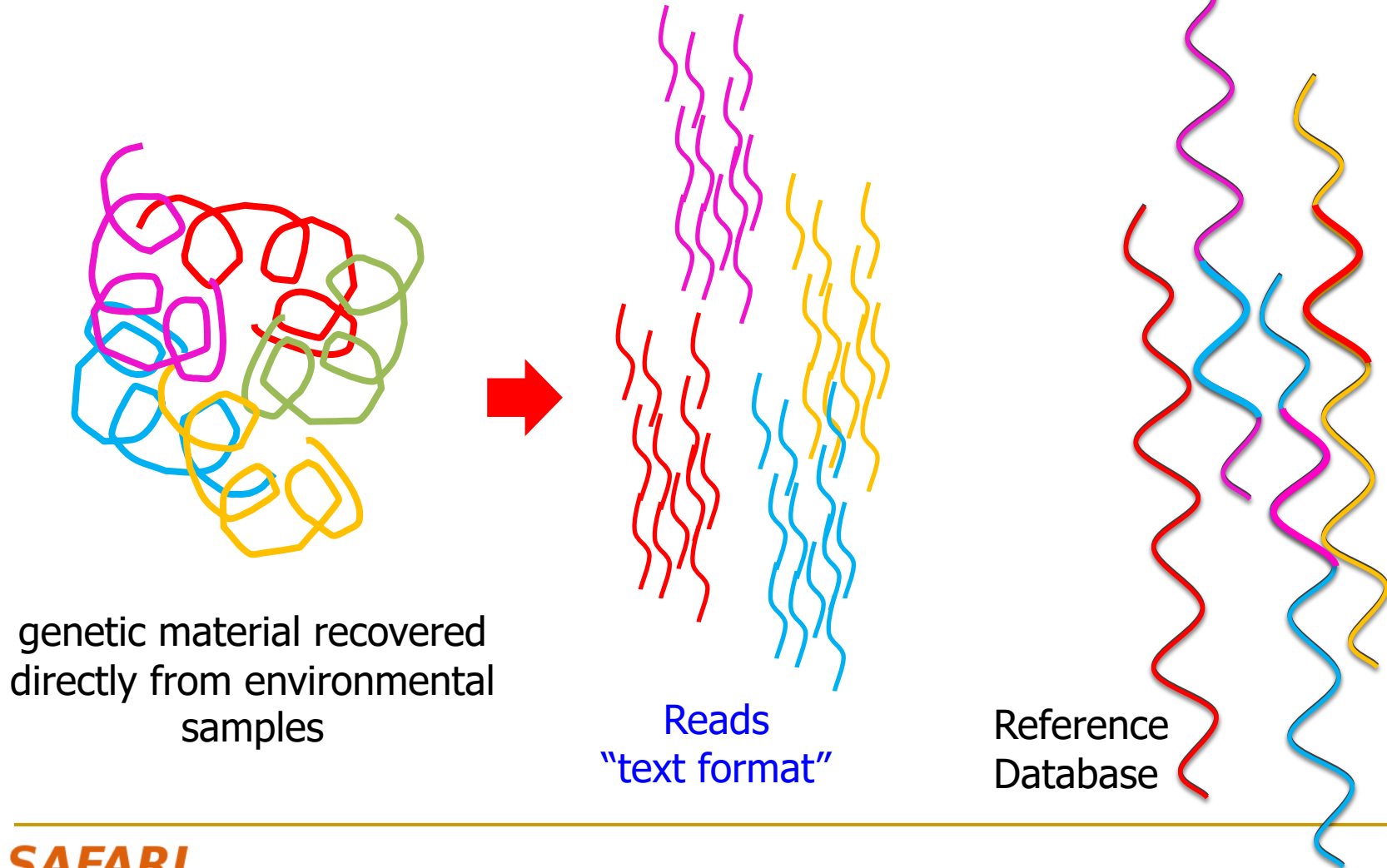
Genome Analysis

Map **reads** to a known reference genome with some minor differences allowed

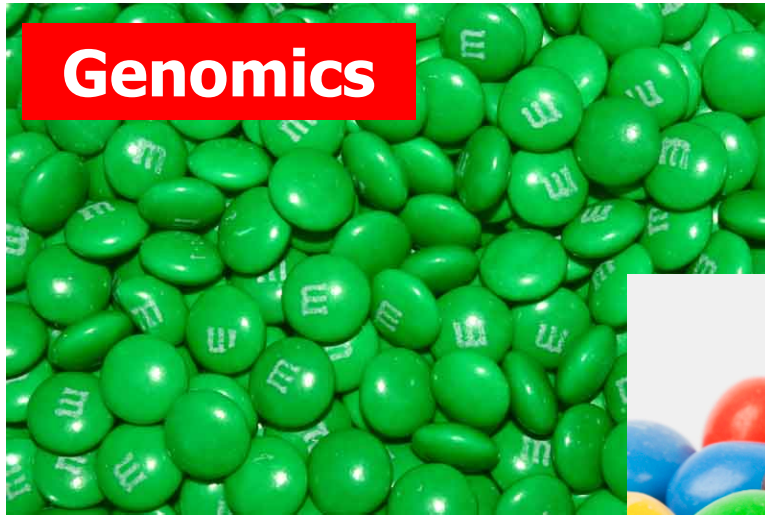


Metagenomics Analysis

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

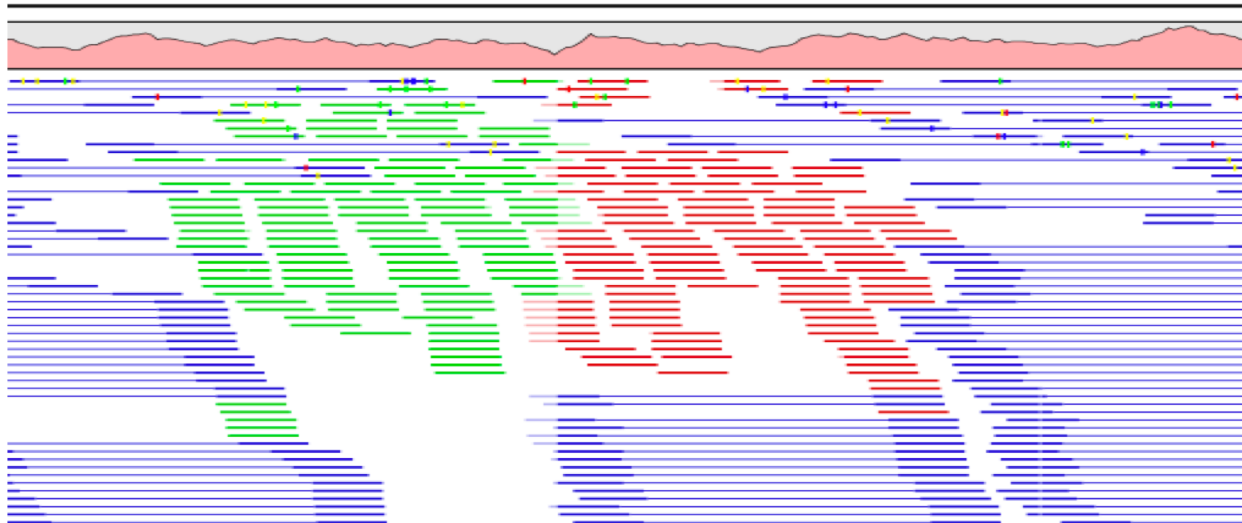


Genomics vs. Metagenomics



Challenges in Read Mapping

- Need to find many **mappings** of **each read**
- Need to **tolerate** small **variances/errors** in each read
- Need to **map** each read **very fast** (i.e., performance is important, life critical in some cases)



Agenda for Today

- Why Genome Analysis?
- What is Genome Analysis?
- How we Map Reads?
- What Makes Read Mapper Slow?
- Algorithmic & Hardware Acceleration
 - Seed Filtering Technique
 - Pre-alignment Filtering Technique
 - Read Alignment Acceleration
- Where is Read Mapping Going Next?

Read Mapping: A Brute Force Algorithm

Reference



Read

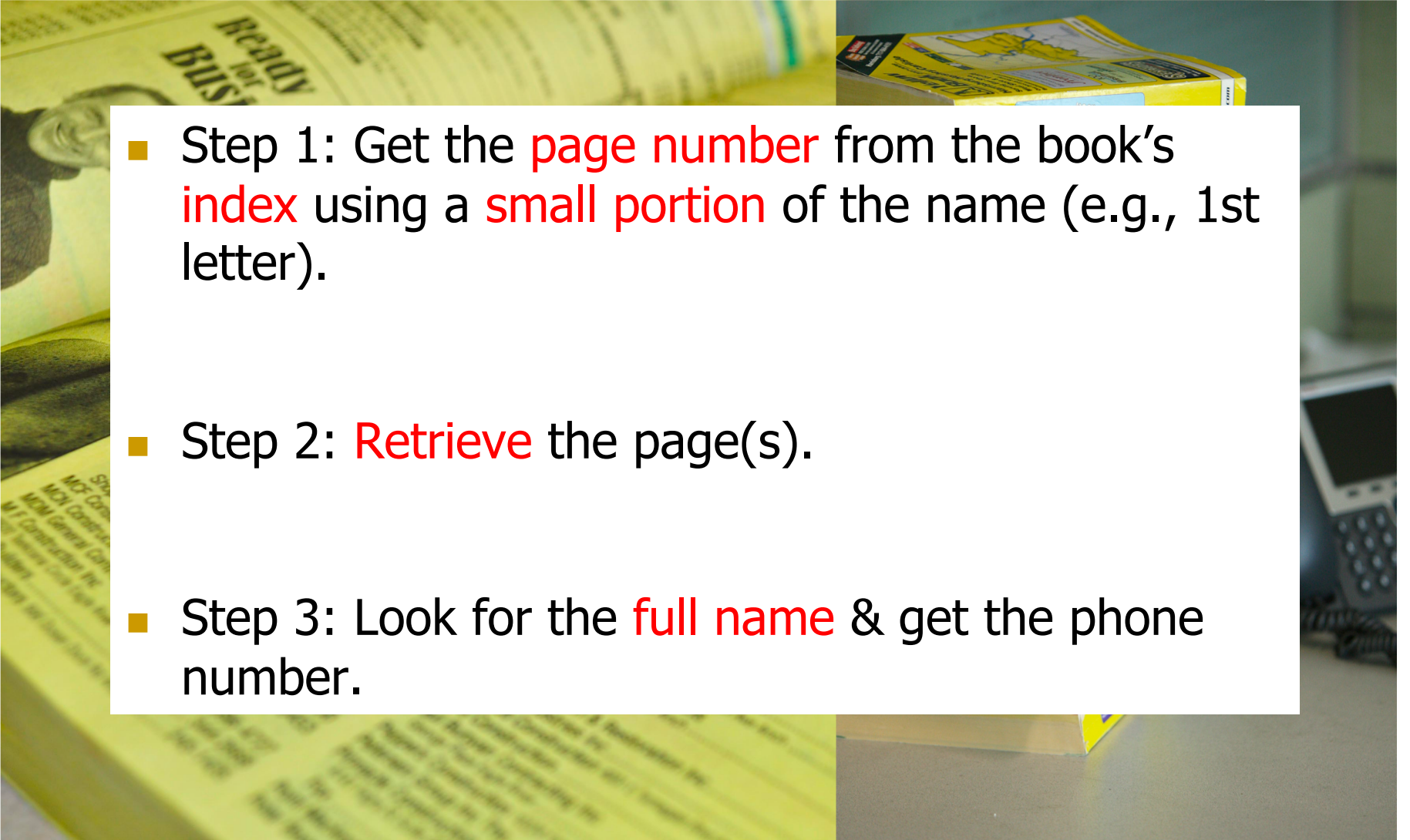
Very Expensive!
 $O(m^2kn)$

m : read length

k : no. of reads

n : reference genome length

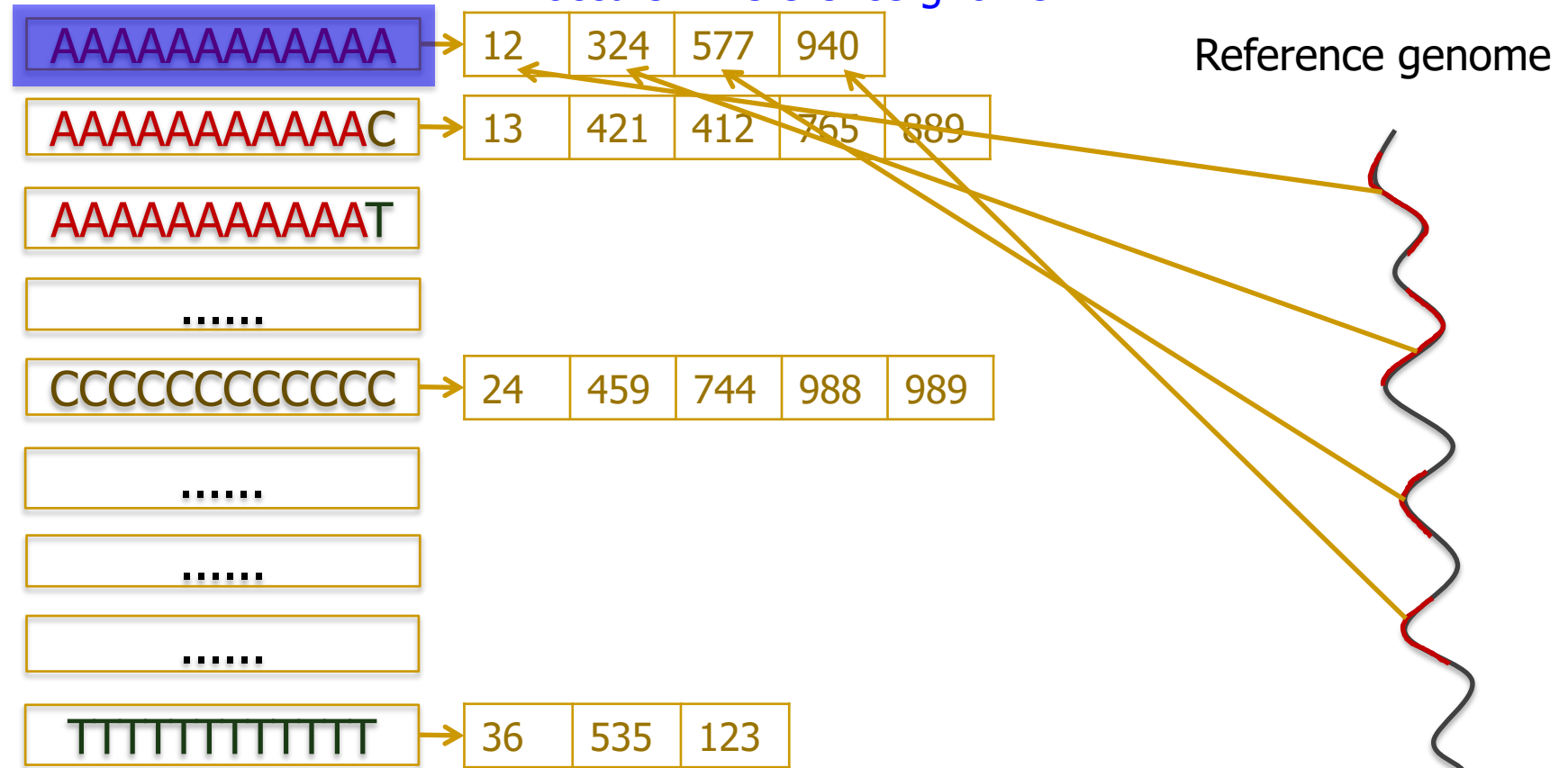
Similar to Searching Yellow Pages!

- 
- 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: Look for the **full name** & get the phone number.

Step 1: Indexing the Reference Genome

Seed=k-mer or 12-mer
(string of length k)

Location list—where the k-mer
occurs in reference genome



We can query the table with substrings from reads
to quickly find a list of possible mapping locations

Genome Index Properties

- The index is built **only once** for each reference.
- **Seeds** can be overlapping, non-overlapping, spaced, adjacent, non-adjacent, minimizers, compressed, ...

Tool	Version	Index Size [*]	Indexing Time
mrFAST	2.2.5	16.5 GB	20.00 min
minimap2	0.12.7	7.2 GB	3.33 min
BWA-MEM	0.7.17	4.7 GB	49.96 min

^{*}Human genome = 3.2 GB

Step 2: Query the Index Using Read Seeds

Read Sequence (100 bp)


Match...


Mismatch.


Mismatch


Mismatch


Mismatch

Reference Genome

Index
(e.g., Hash Table)

37 140
894 1203
1564

Step 3: Read 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.

organization x operation

Ref	o	-	-	r	g	a	n	i	z	a	t	i	o	n
Read	o	p	e	r	-	-	-	-	-	a	t	i	o	n

Ref	o	-	-	r	g	a	n	i	z	a	t	i	o	n
Read	o	p	e	r	-	a	-	-	-	-	t	i	o	n

Edit distance = 7

match
deletion
insertion
mismatch

organization x translation

Ref	o	r	g	a	n	i	z	-	a	t	i	o	n
Read	t	r	-	a	n	-	s	-	a	t	i	o	n

Ref	o	r	g	a	n	-	i	z	a	t	i	o	n
Read	t	r	-	a	n	s	-	-	a	t	i	o	n

Ref	o	r	g	a	n	i	z	a	t	i	o	n
Read	t	r	-	a	n	s	-	a	t	i	o	n

Edit distance = 4

An Example of Hash Table Based Mappers

- + Guaranteed to find *a//* mappings → very sensitive
- + Can tolerate up to *e* errors

nature
genetics

<https://github.com/BilkentCompGen/mrfast>

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

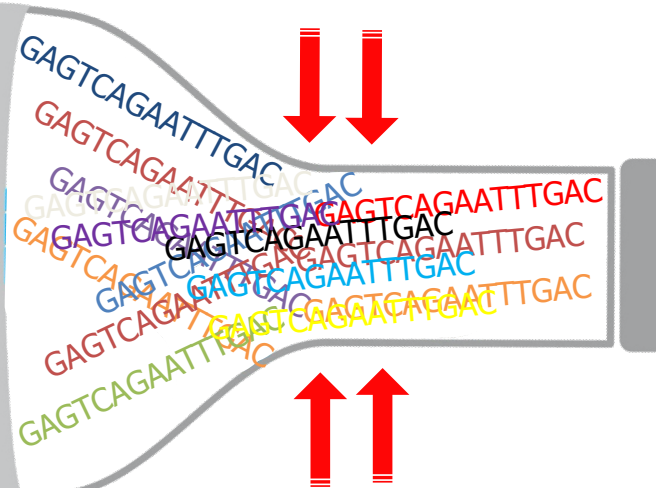
Can Alkan^{1,2}, Jeffrey M Kidd¹, Tomas Marques-Bonet^{1,3}, Gozde Aksay¹, Francesca Antonacci¹, Fereydoun Hormozdiari⁴, Jacob O Kitzman¹, Carl Baker¹, Maika Malig¹, Onur Mutlu⁵, S Cenk Sahinalp⁴, Richard A Gibbs⁶ & Evan E Eichler^{1,2}

SAFARI Alkan+, "Personalized copy number and segmental duplication maps using next-generation sequencing", Nature Genetics 2009.

Bottlenecked in Read Alignment!!

378 Million
bases/minute

Read Sequencing **



2 Million
bases/minute

Read Mapping *

150x slower

* BWA-MEM

** NovaSeq 6000, MinION

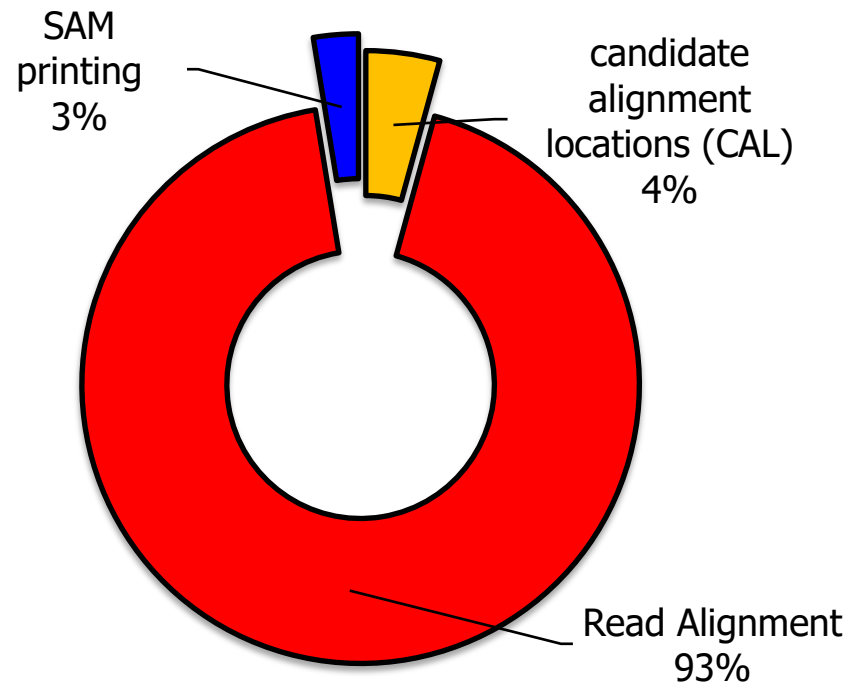
Agenda for Today

- Why Genome Analysis?
- What is Genome Analysis?
- How we Map Reads?
- What Makes Read Mapper Slow?
- Algorithmic & Hardware Acceleration
 - Seed Filtering Technique
 - Pre-alignment Filtering Technique
 - Read Alignment Acceleration
- Where is Read Mapping Going Next?

What Makes Read Mapper Slow?

Key Observation # 1

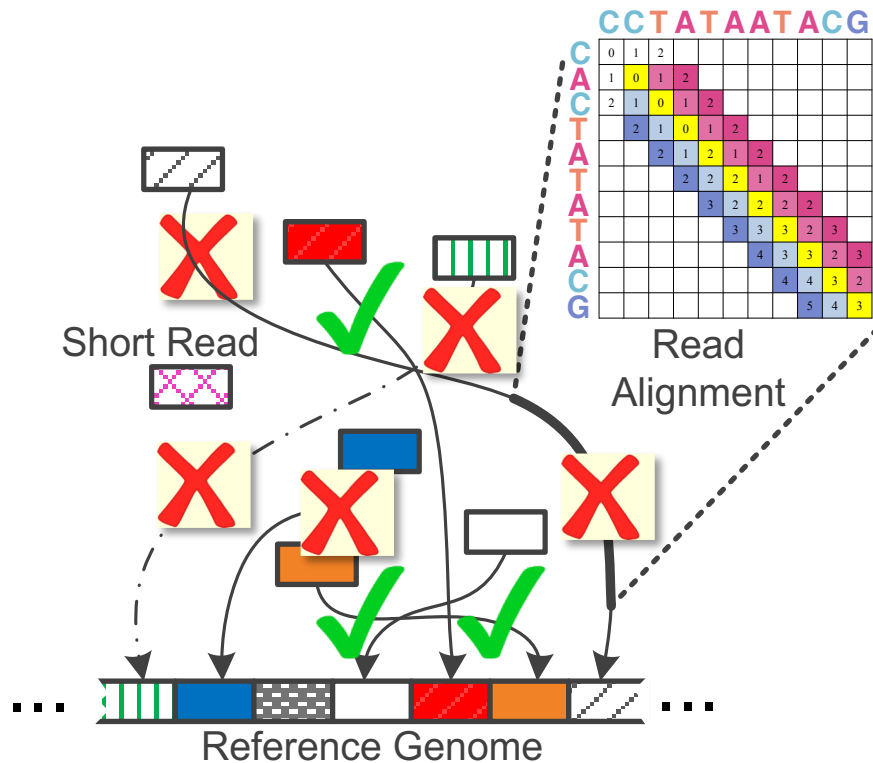
93%
of the read mapper's
execution time is spent
in read alignment.



Alser et al, Bioinformatics (2017)

What Makes Read Mapper Slow? (cont'd)

Key Observation # 2



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

Cheng *et al*, *BMC bioinformatics* (2015)
Xin *et al*, *BMC genomics* (2013)

What Makes Read Mapper Slow? (cont'd)

Key Observation # 3

- **Quadratic-time** dynamic-programming algorithm **WHY?!**

Enumerating all possible prefixes

- 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

		N	E	T	H	E	R	L	A	N	D	S	
		0	1	2	3	4	5	6	7	8	9	10	11
S	1												
W	2												
I	3												
T	4												
Z	5												
E	6												
R	7												
L	8												
A	9												
N	10												
D	11												

What Makes Read Mapper Slow? (cont'd)

Key Observation # 3

- **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 backtraking step.

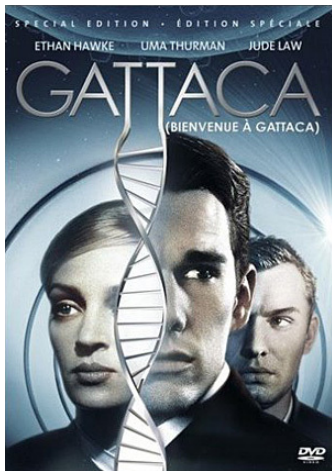
		N	E	T	H	E	R	L	A	N	D	S
	0	1	2	3	4	5	6	7	8	9	10	11
S	1	1	2	3	4	5	6	7	8	9	10	10
W	2	2	2	3	4	5	6	7	8	9	10	11
I	3	3	3	3	4	5	6	7	8	9	10	11
T	4	4	4	3	4	5	6	7	8	9	10	11
Z	5	5	5	4	4	5	6	7	8	9	10	11
E	6	6	5	5	5	4	5	6	7	8	9	10
R	7	7	6	6	6	5	4	5	6	7	8	9
L	8	8	7	7	7	6	5	4	5	6	7	8
A	9	9	8	8	8	7	6	5	4	5	6	7
N	10	9	9	9	9	8	7	6	5	4	5	6
D	11	10	10	10	10	9	8	7	6	5	4	5

Agenda for Today

- Why Genome Analysis?
- What is Genome Analysis?
- How we Map Reads?
- What Makes Read Mapper Slow?
- Algorithmic & Hardware Acceleration
 - Seed Filtering Technique
 - Pre-alignment Filtering Technique
 - Read Alignment Acceleration
- Where is Read Mapping Going Next?

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).



1997

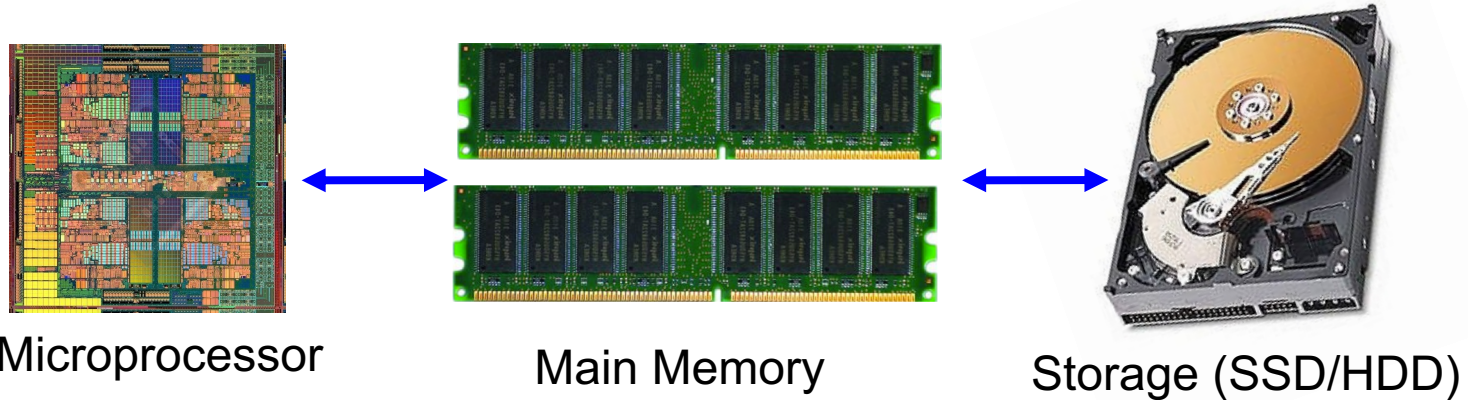


2015

Open Questions

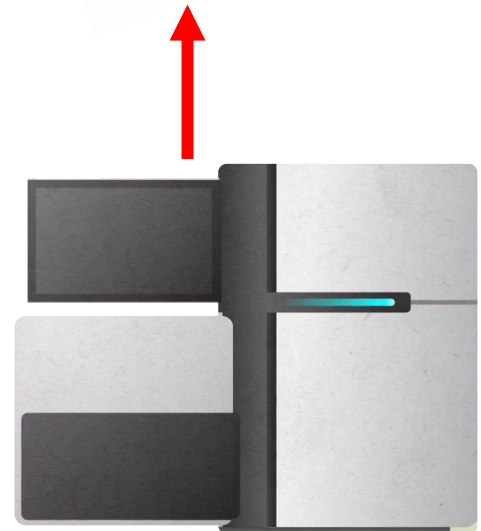
How and where to enable
fast, accurate, cheap,
privacy-preserving, and exabyte scale
analysis of genomic data?

Pushing Towards New Architectures

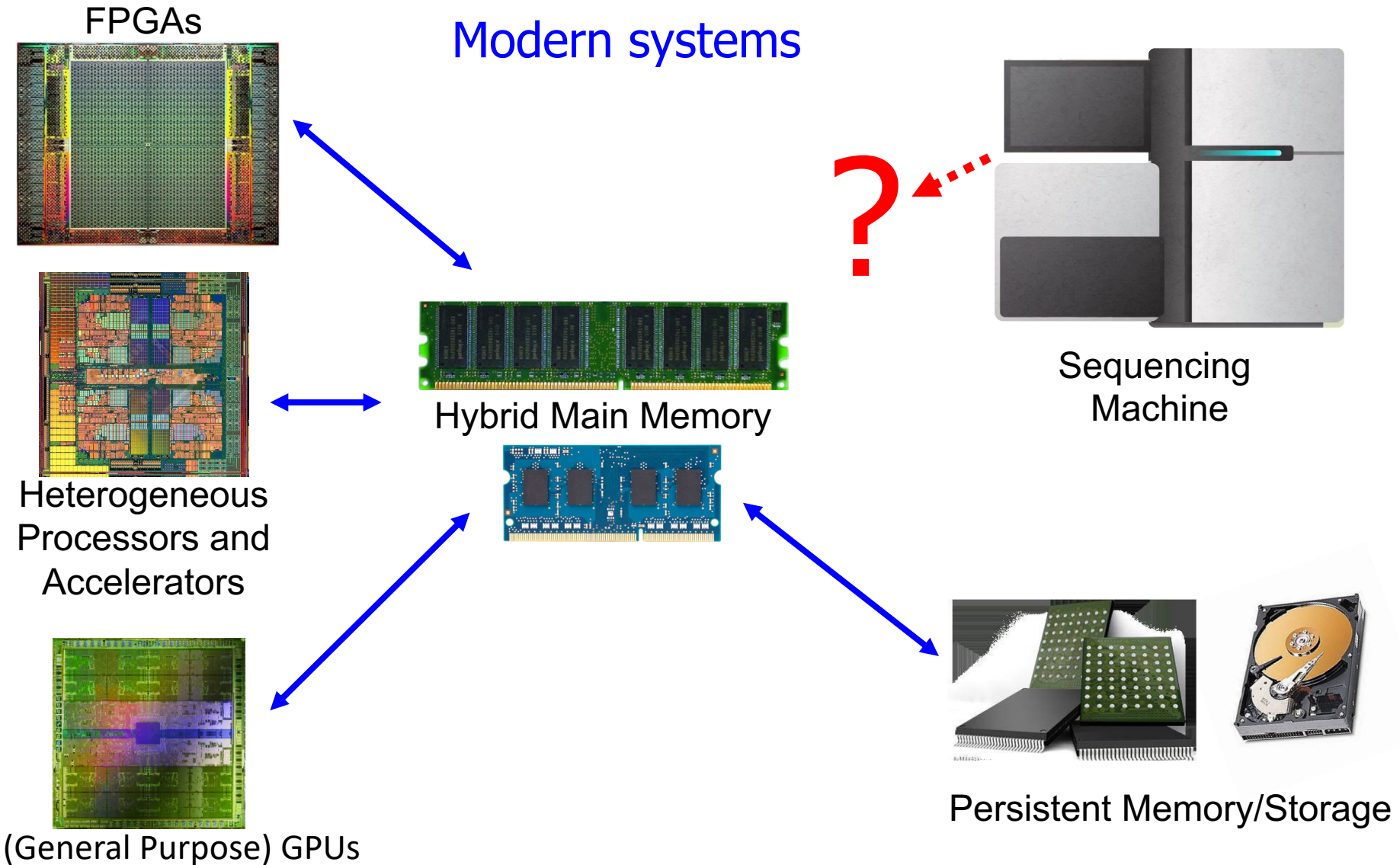


Single **memory** request **consumes**
>160x-800x **more energy** compared to
performing a **complex add operation**

Sequencing
Machine

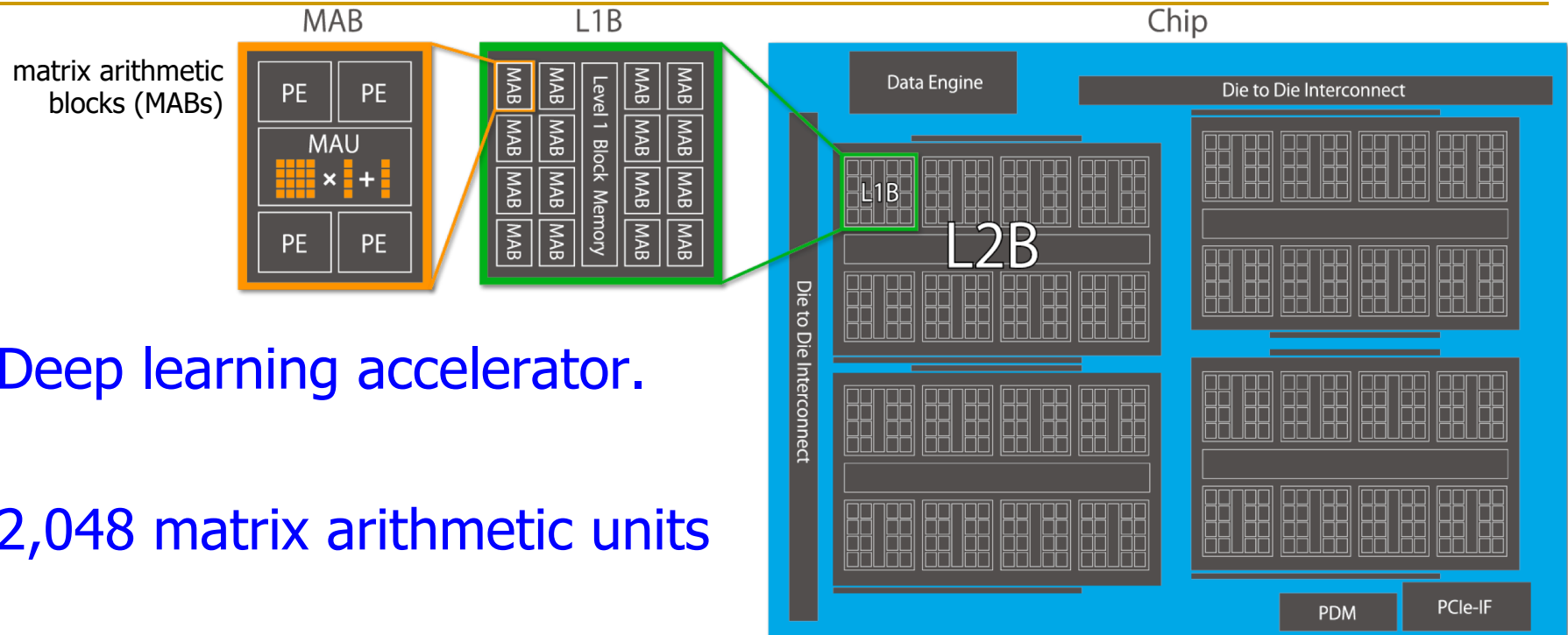


Processing Genomic Data Where it Makes Sense



Most speedup comes from **parallelism** enabled
by **novel architectures** and **algorithms**

Preferred Networks' MN-Core (2018)

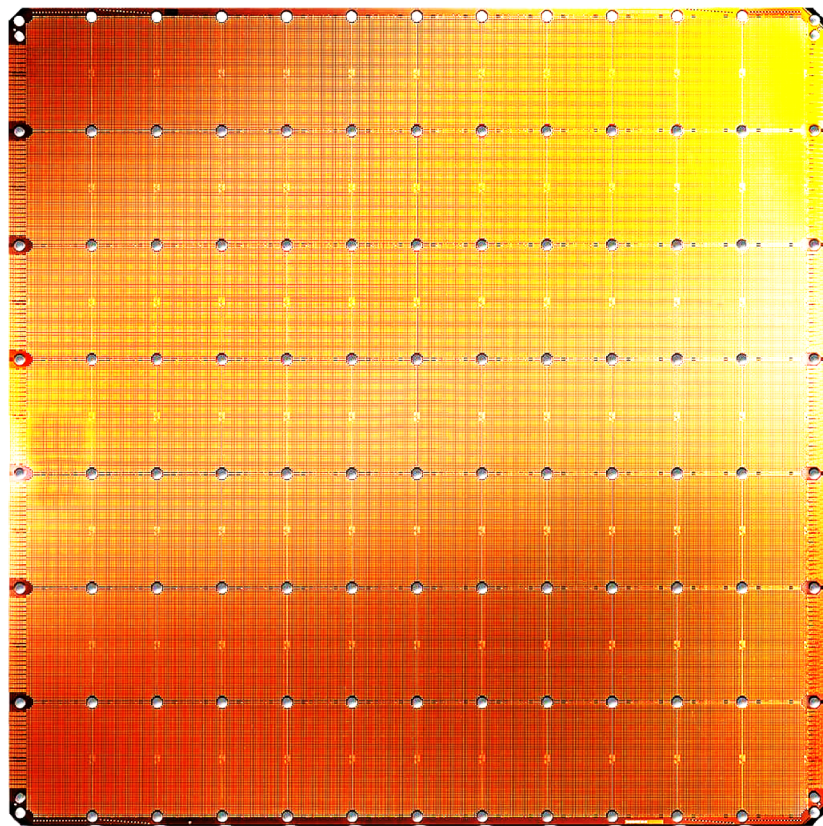


- Deep learning accelerator.
- 2,048 matrix arithmetic units

Fabrication process	TSMC 12nm
Estimated power consumption (W)	500
Peak performance (TFLOPS)	32.8 (DP) / 131 (SP) / 524 (HP)
Estimated performance per watt (TFLOPS / W)	0.066 (DP) / 0.26 (SP) / 1.0 (HP)

(Notes) DP: double precision, SP: single precision, HP: half precision.

Cerebras's Wafer Scale Engine (2019)



Cerebras WSE

1.2 Trillion transistors

46,225 mm²

- The largest ML accelerator chip
- 400,000 cores



Largest GPU

21.1 Billion transistors

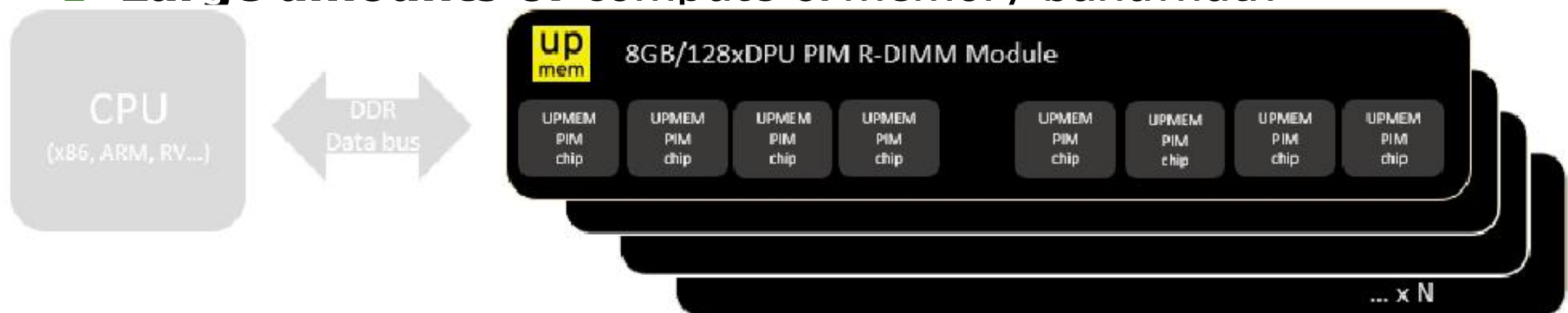
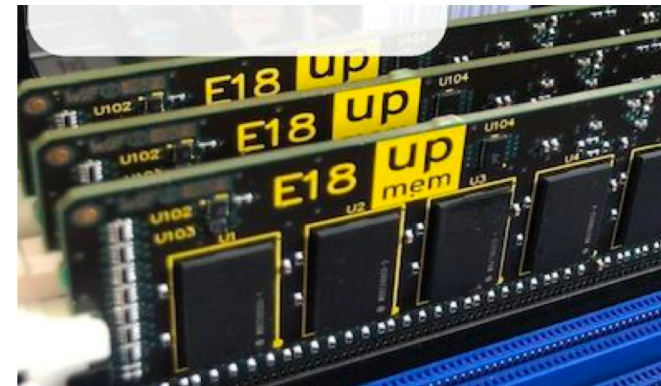
815 mm²

NVIDIA TITAN V

<https://www.cerebras.net/cerebras-wafer-scale-engine-why-we-need-big-chips-for-deep-learning/>

UPMEM Processing-in-DRAM Engine (2019)

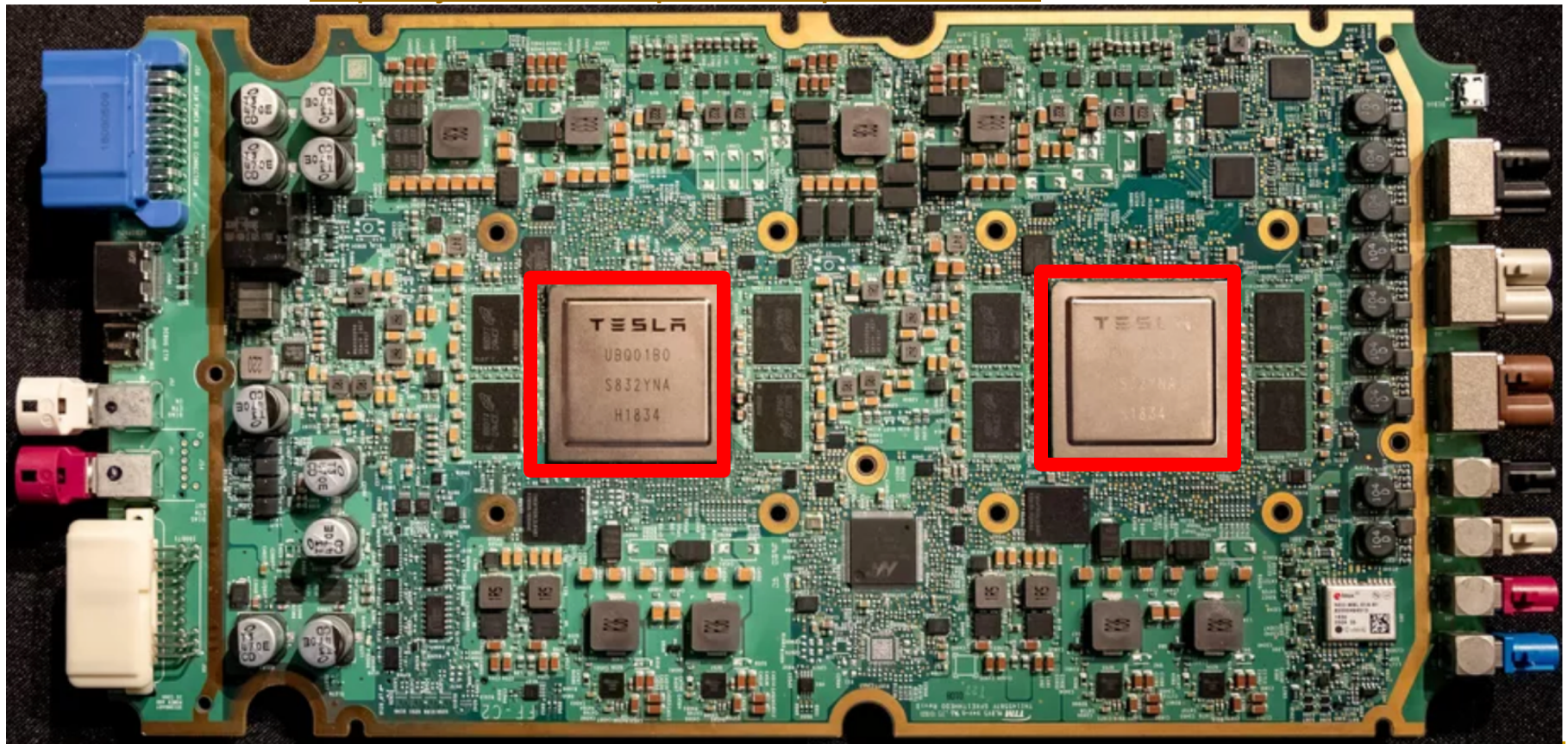
- **Processing in DRAM Engine**
- Includes **standard DIMM modules**, with a **large number of DPU processors** combined with DRAM chips.
- Replaces **standard DIMMs**
 - DDR4 R-DIMM modules
 - 8GB+128 DPUs (16 PIM chips)
 - Standard 2x-nm DRAM process
 - **Large amounts of** compute & memory bandwidth



TESLA Full Self-Driving Computer (2019)

- ML accelerator: 260 mm², 6 billion transistors, 600 GFLOPS GPU, 12 ARM 2.2 GHz CPUs.
- Two redundant chips for better safety.

<https://youtu.be/Ucp0TTmvqOE?t=4236>



Illumina + Edico Genome

PRESS RELEASE

Illumina Acquires Edico Genome to Accelerate Genomic Data Analysis

Edico's DRAGEN® Bio-IT Platform Delivers Faster, Streamlined Output for Next-Generation Sequencing

SAN DIEGO--(BUSINESS WIRE)--May 15, 2018-- [Illumina, Inc.](#) (NASDAQ: ILMN) today announced that it acquired Edico Genome, the leading provider of data analysis acceleration solutions for next-generation sequencing (NGS). Edico Genome's DRAGEN® Bio-IT Platform (DRAGEN) uses field programmable gate array (FPGA) technology in conjunction with proprietary software algorithms to reduce both data footprint and time to results.

<https://www.illumina.com/company/news-center/press-releases/2018/2349147.html>

Illumina + PacBio

PRESS RELEASE

Illumina to Acquire Pacific Biosciences for Approximately \$1.2 Billion, Broadening Access to Long-Read Sequencing and Accelerating Scientific Discovery

- *Brings Together Highly Accurate Short- and Long-Read Sequencing Technologies, Paving the Path to a More Perfect View of a Genome*
- *Pacific Biosciences' Recent Advances with its Sequel SMRT® Technology, Combined with Illumina's Infrastructure, will Expand Biological Discovery and Clinical Insight*
- *Long-Read Sequencing Market Opportunity Expected to Grow to \$2.5B by 2022*

SAN DIEGO & MENLO PARK, Calif.--(BUSINESS WIRE)--Nov. 1, 2018-- Illumina, Inc. (NASDAQ: ILMN)

<https://www.illumina.com/company/news-center/press-releases/press-release-details.html?newsid=2374913>

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.

An Example of Ongoing Directions

Read Sequence (100 bp)



2) Pre-Alignment Filtering...

↓ **Match!**

3) Rapid Alignment

Matching...

Hash Table

37 140
894 1203
1564

**Pre-Alignment
Mismatch
Filtering...**

Reference Genome

1) Seed Filtering ...

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.

FastHASH

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

Xin *et al.* *BMC Genomics* 2013, **14**(Suppl 1):S13
<http://www.biomedcentral.com/1471-2164/14/S1/S13>



PROCEEDINGS

Open Access

Accelerating read mapping with FastHASH

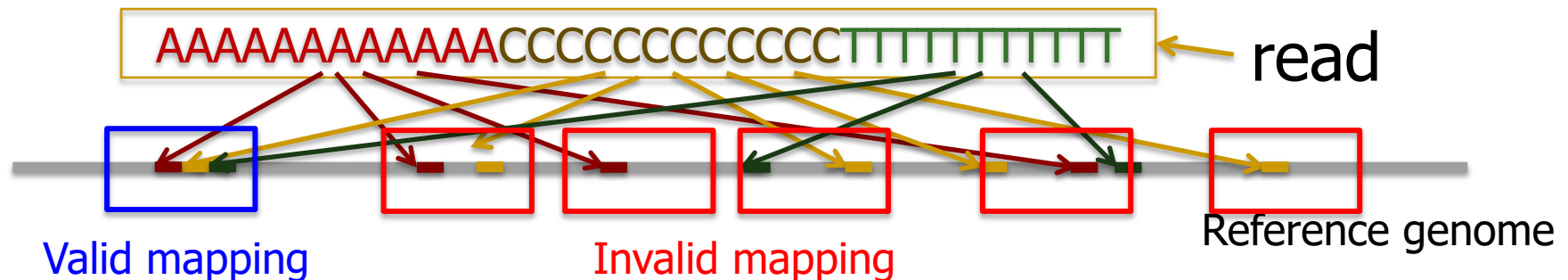
Hongyi Xin¹, Donghyuk Lee¹, Farhad Hormozdiari², Samihan Yedkar¹, Onur Mutlu^{1*}, Can Alkan^{3*}

From The Eleventh Asia Pacific Bioinformatics Conference (APBC 2013)
Vancouver, Canada. 21-24 January 2013

Key Observations

■ Observation 1 (Adjacent k-mers)

- ❑ **Key insight:** Adjacent k-mers in the read should also be adjacent in the reference genome
- ❑ **Key idea:** 1) sort the location list based on their number of locations and 2) search for adjacent locations in the k-mers' location lists



Key Observations

■ Observation 1 (Adjacent k-mers)

- ❑ **Key insight:** Adjacent k-mers in the read should also be adjacent in the reference genome
- ❑ **Key idea:** 1) sort the location list based on their number of locations and 2) search for adjacent locations in the k-mers' location lists

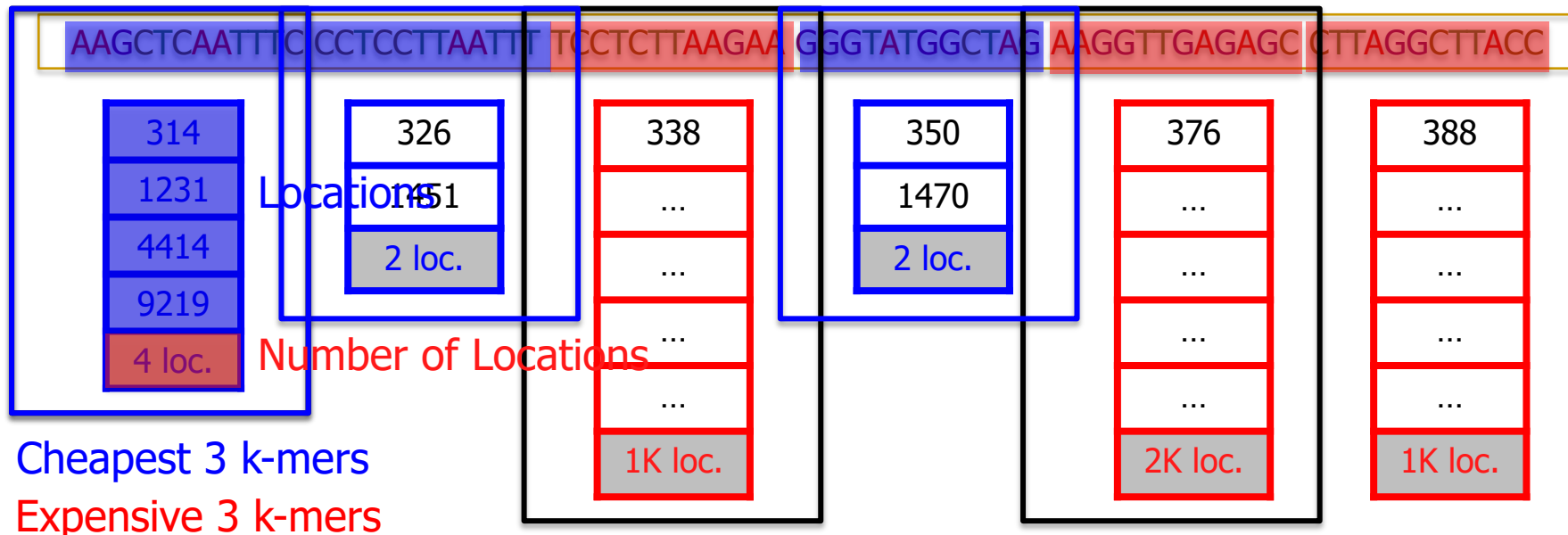
■ Observation 2 (Cheap k-mers)

- ❑ **Key insight:** Some k-mers are cheaper to verify than others because they have shorter location lists (they occur less frequently in the reference genome)
- ❑ **Key Idea:** Read mapper can choose the cheapest k-mers and verify their locations

Cheap K-mer Selection

- occurrence threshold = 500

read



Previous work needs to verify:

3004 locations



FastHASH verifies only:

8 locations

FastHASH Conclusion

- **Problem:** Existing **read mappers** perform **poorly** in mapping billions of short reads to the reference genome, in the presence of errors
- **Observation:** Most of the **verification** calculations are unnecessary → filter them out
- **Key Idea:** To reduce the cost of unnecessary verification
 - ❑ Select **Cheap** and **Adjacent** k-mers.
- **Key Result:** FastHASH obtains up to **19x** speedup over the state-of-the-art mapper without losing valid mappings

More on FastHASH

- Download source code and try for yourself
 - [Download link to FastHASH](#)

Xin *et al.* *BMC Genomics* 2013, **14**(Suppl 1):S13
<http://www.biomedcentral.com/1471-2164/14/S1/S13>



PROCEEDINGS

Open Access

Accelerating read mapping with FastHASH

Hongyi Xin¹, Donghyuk Lee¹, Farhad Hormozdiari², Samihan Yedkar¹, Onur Mutlu^{1*}, Can Alkan^{3*}

From The Eleventh Asia Pacific Bioinformatics Conference (APBC 2013)
Vancouver, Canada. 21-24 January 2013

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:**

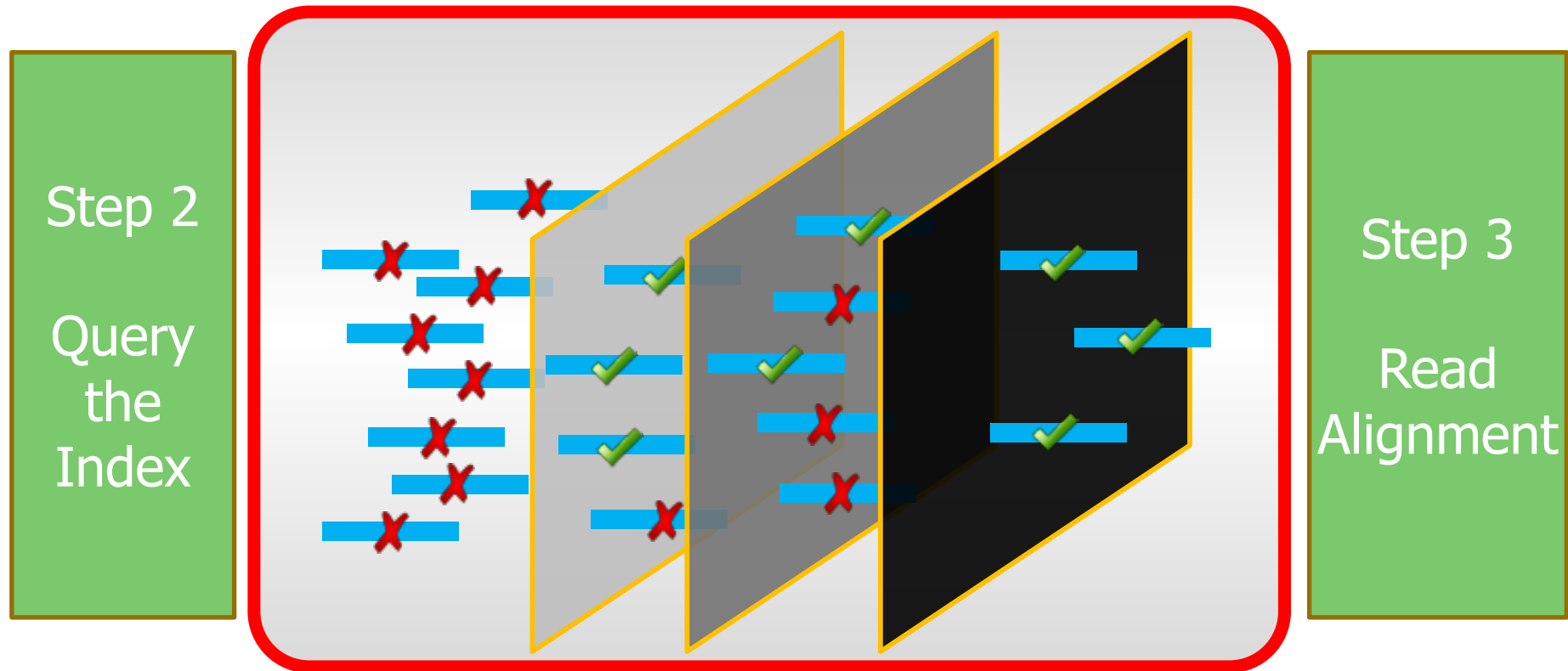
- **Goal:** Performing read alignment at scale.
 - Limits the **numeric range** of each cell in the DP table and hence supports **limited scoring** function.
 - May not support **backtracking** step due to random memory accesses.

Pre-alignment Filtering Technique

Read Alignment is expensive

Our goal is to reduce the need for dynamic programming algorithms

Ideal Filtering Algorithm



1. **Filter out** most of incorrect mappings.
2. **Preserve** all correct mappings.
3. Do it **quickly**.

Bioinformatics



Article Navigation

GateKeeper: a new hardware architecture for accelerating pre-alignment in DNA short read mapping ^{FREE}

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

Bioinformatics, Volume 33, Issue 21, 01 November 2017, Pages 3355–3363,

<https://doi.org/10.1093/bioinformatics/btx342>

Published: 31 May 2017 **Article history** ▼

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

GateKeeper

■ Key observation:

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

■ Key idea:

- Compute “Shifted Hamming Distance”: AND of $2E+1$ Hamming vectors of two strings, to identify invalid mappings
 - Uses *bit-parallel operations* that nicely map to FPGA architectures

■ Key result:

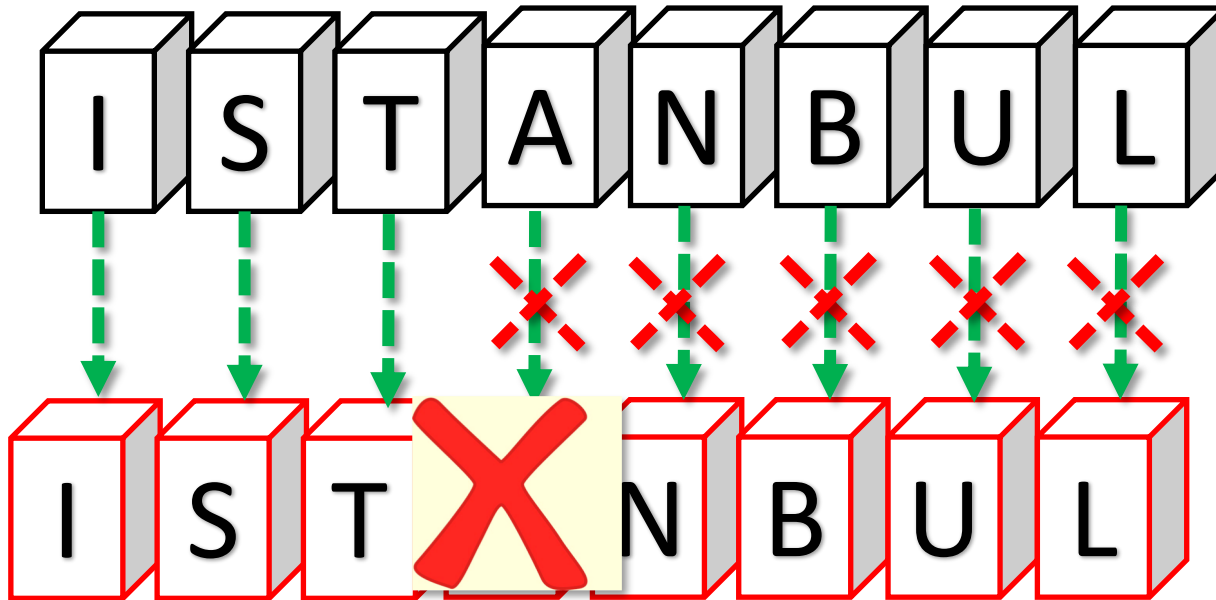
- GateKeeper is 90x-130x faster than SHD (Xin et al., 2015) and the Adjacency Filter (Xin et al., 2013), with only a 7% false positive rate
- The addition of GateKeeper to the mrFAST mapper (Alkan et al., 2009) results in 10x end-to-end speedup in read mapping

Hamming Distance ($\Sigma \oplus$)

3 matches

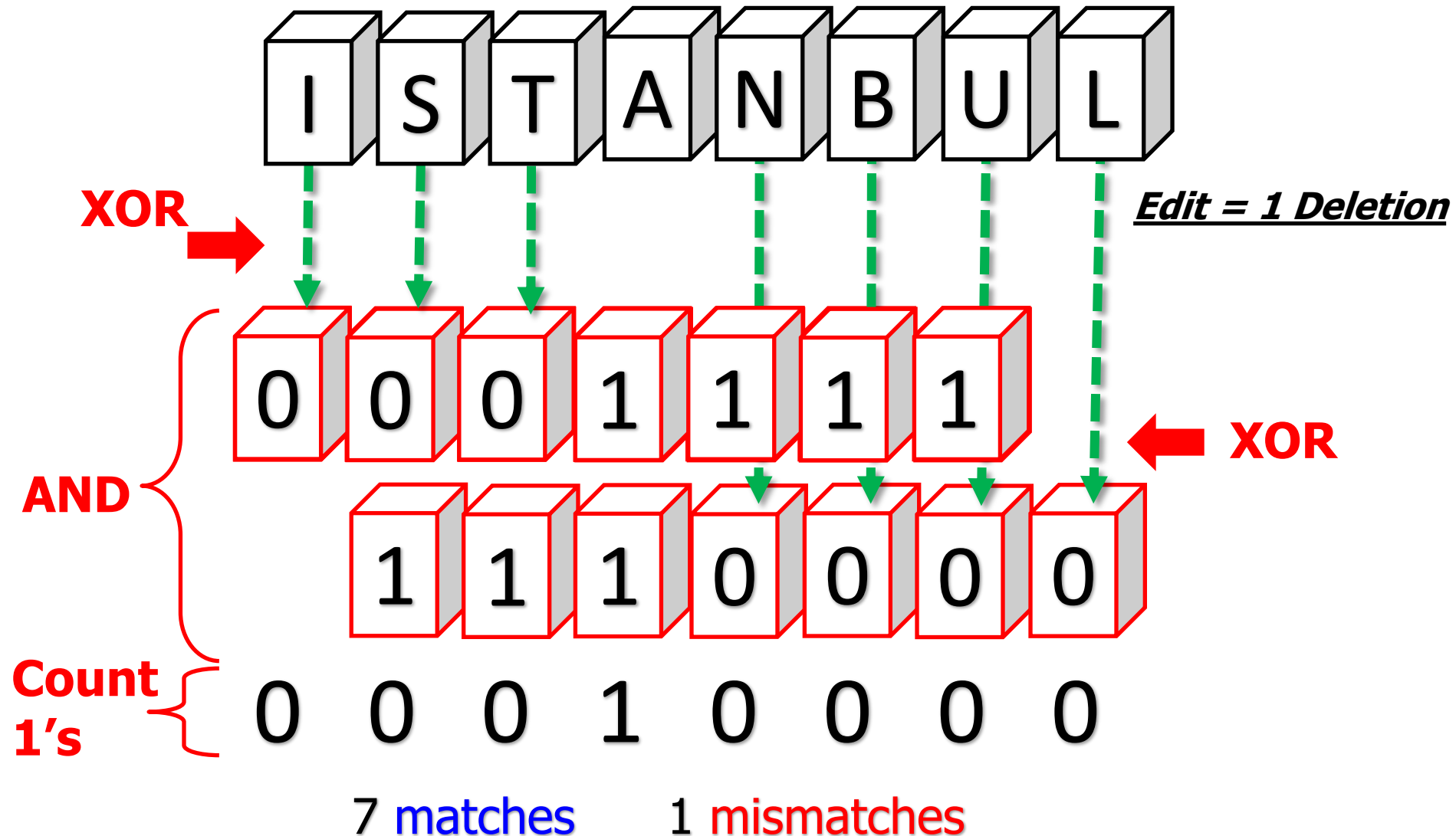
5 mismatches

Edit = 1 Deletion



To cancel the effect of a deletion, we need to shift in the *right* direction

Shifted Hamming Distance (Xin+ 2015)



GateKeeper Walkthrough

Generate $2E+1$ masks

Amend random zeros:
101 → 111 & 1001 → 1111

AND all masks,
ACCEPT iff number of '1' \leq Threshold

Query :GAGAGAGATATTTAGTGTTGCAGCACTACAACACAAAAGAGGACCAACTTACGTGTCTAAAAGGGGGAACATTGTTGGGCCGGA

Reference :GAGAGAGATAGTTAGTGTTCAGCCACTACAACACAAAAGAGGACCAACTTACGTGTCTAAAAGGGGAGACATTGTTGGGCCGG

Hamming Mask : 000000000010000000000011111110111100011101101011011111111000100001111110110110010101

[illegible]

2-Deletion Mask : 000000001011011100111111111111111011110001110110101101111111111000100100111101101001010

3-Deletion Mask :11111111111101110110011011101110110001001001111111111111100101100110101101110111011101111

```
L-Insertion Mask :111111111110111110111110111011000100100111111111111111100101100110000101111011101111110
```

2-Insertion Mask :000000010011111100111111111100100011010101001101011111111111110111001111110001111011000111101100

3-Insertion Mask :111111111011101100110001111111111010110111111001100101110111111110110111010111010111001000

--- Masks after amendment ---

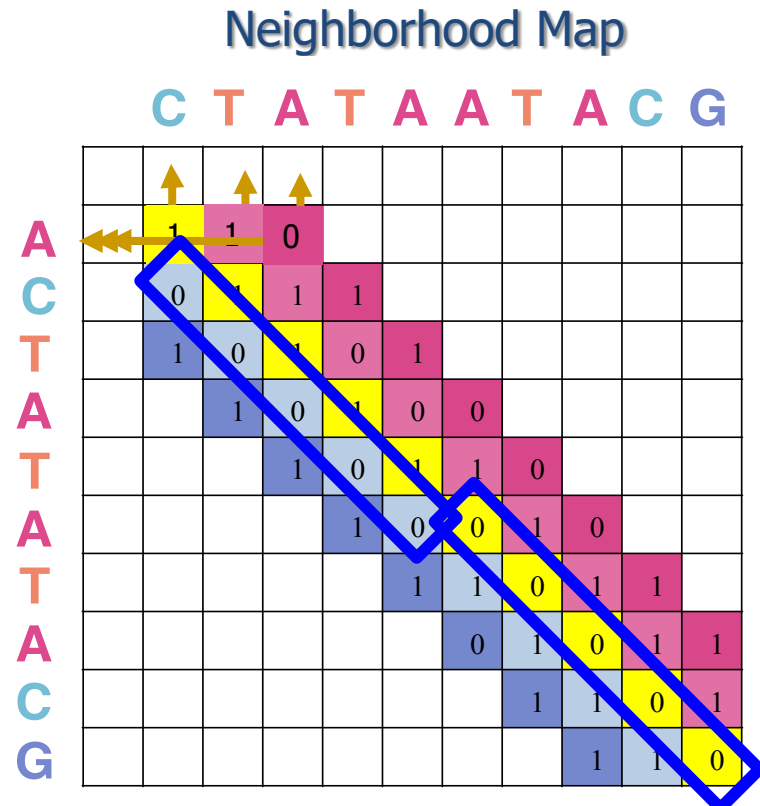
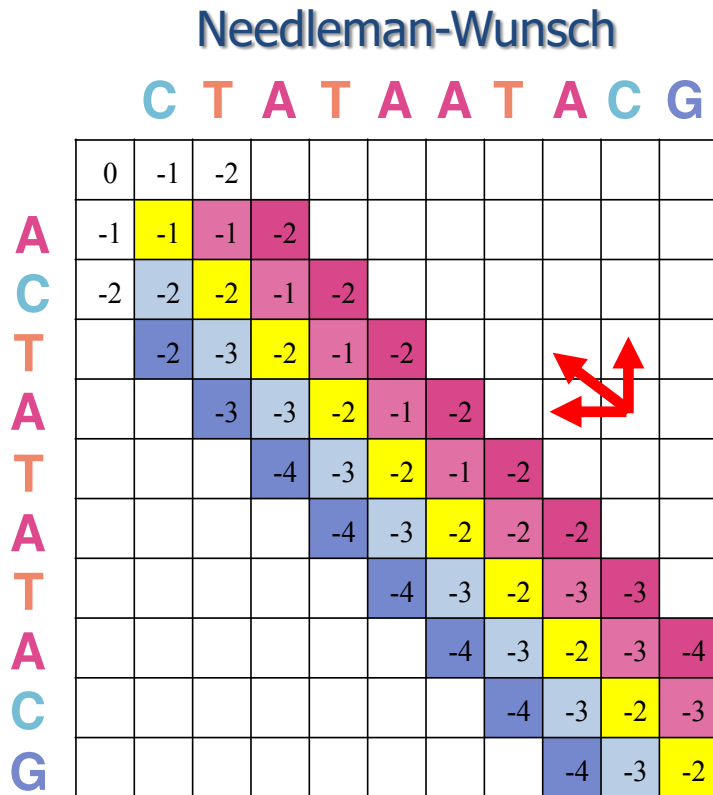
Our goal to track the diagonally consecutive matches in the neighborhood map.

.GAGAGAGATATTTAGTGTTGCAG-CACTACAACACAAAAGAGGACCAACTTACGTGTCTAAAAGGGGGAACATTGTTGGCCCG

[illegible]

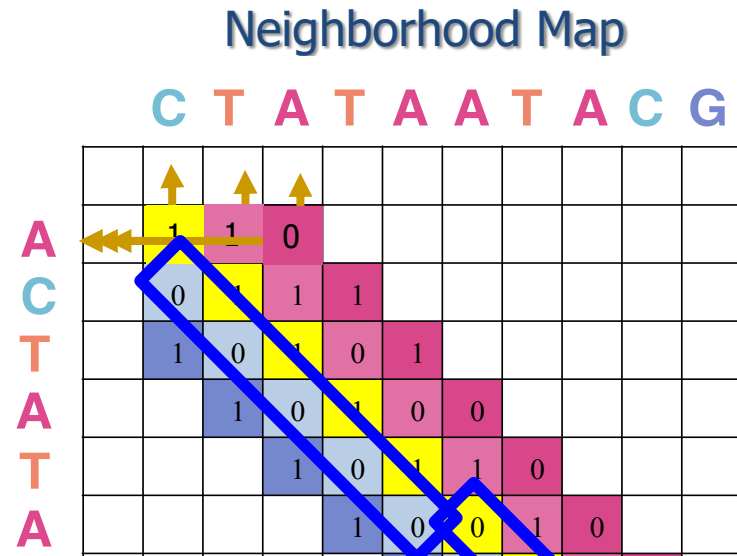
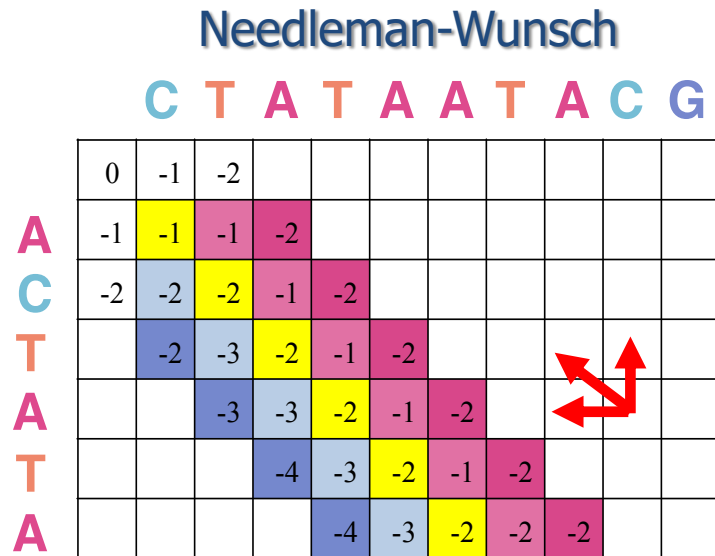
GAGAGAGATAGTTAGTGTTGCAGCCACTACAACACAAAAGAGGACCAACTTACGTGTCTAAAAGGGGAGACATTGTTGGGCCGG

Alignment Matrix vs. Neighborhood Map

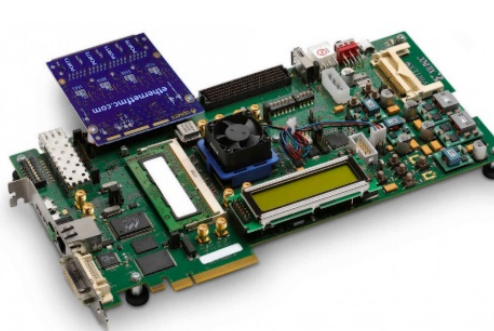


Our goal to track the diagonally consecutive matches in the neighborhood map.

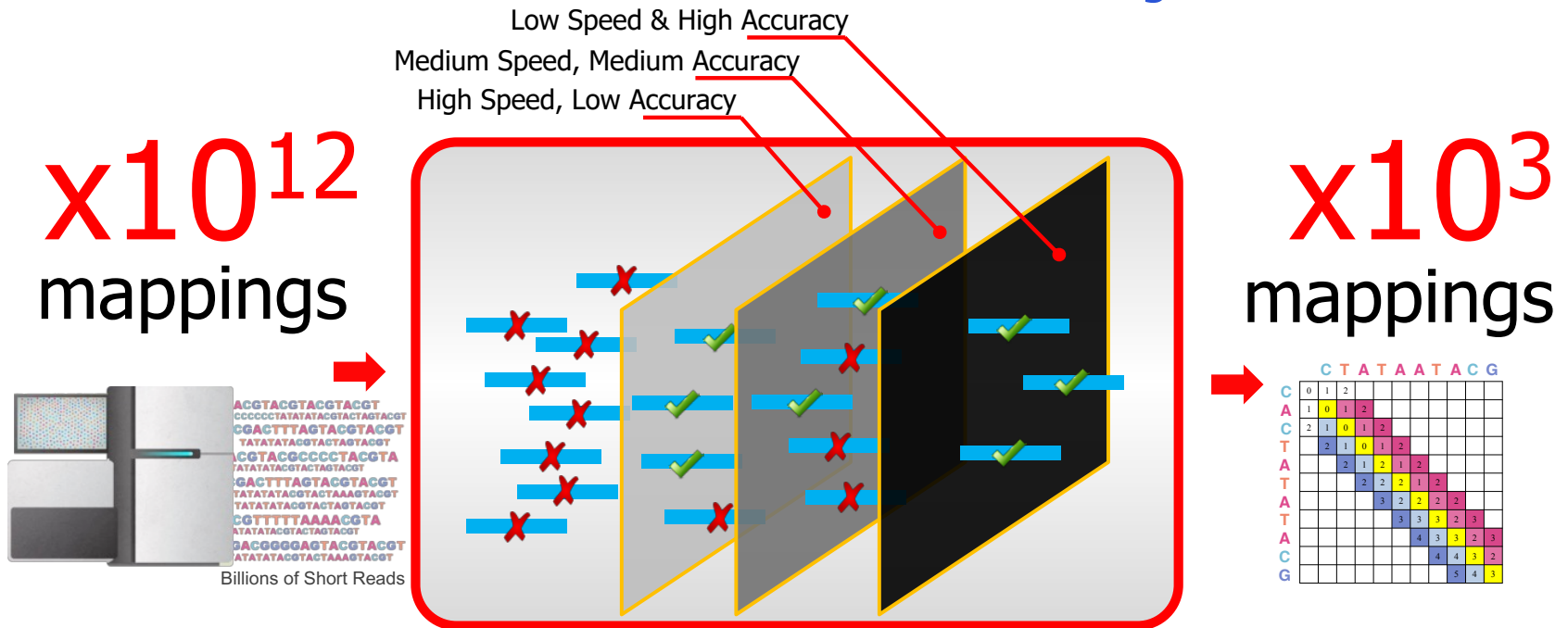
Alignment Matrix vs. Neighborhood Map



Independent vectors can be processed in parallel using hardware technologies



Our Solution: GateKeeper



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

GateKeeper Walkthrough (cont'd)

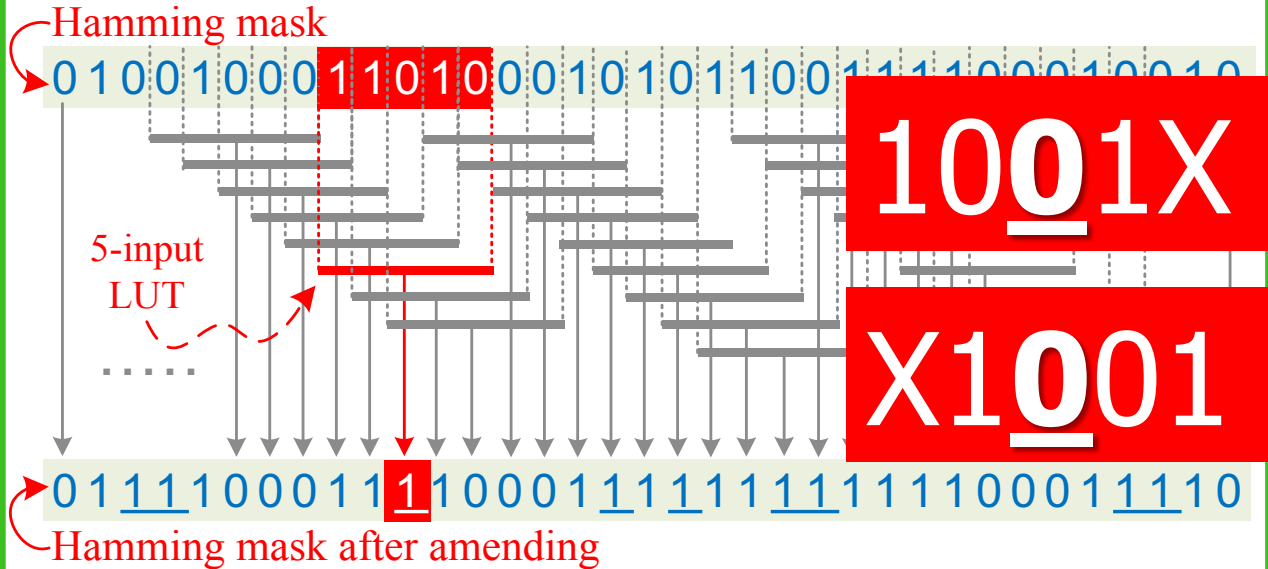
Generate $2E+1$ masks

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

AND all masks,
ACCEPT iff number of '1' \leq Threshold

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

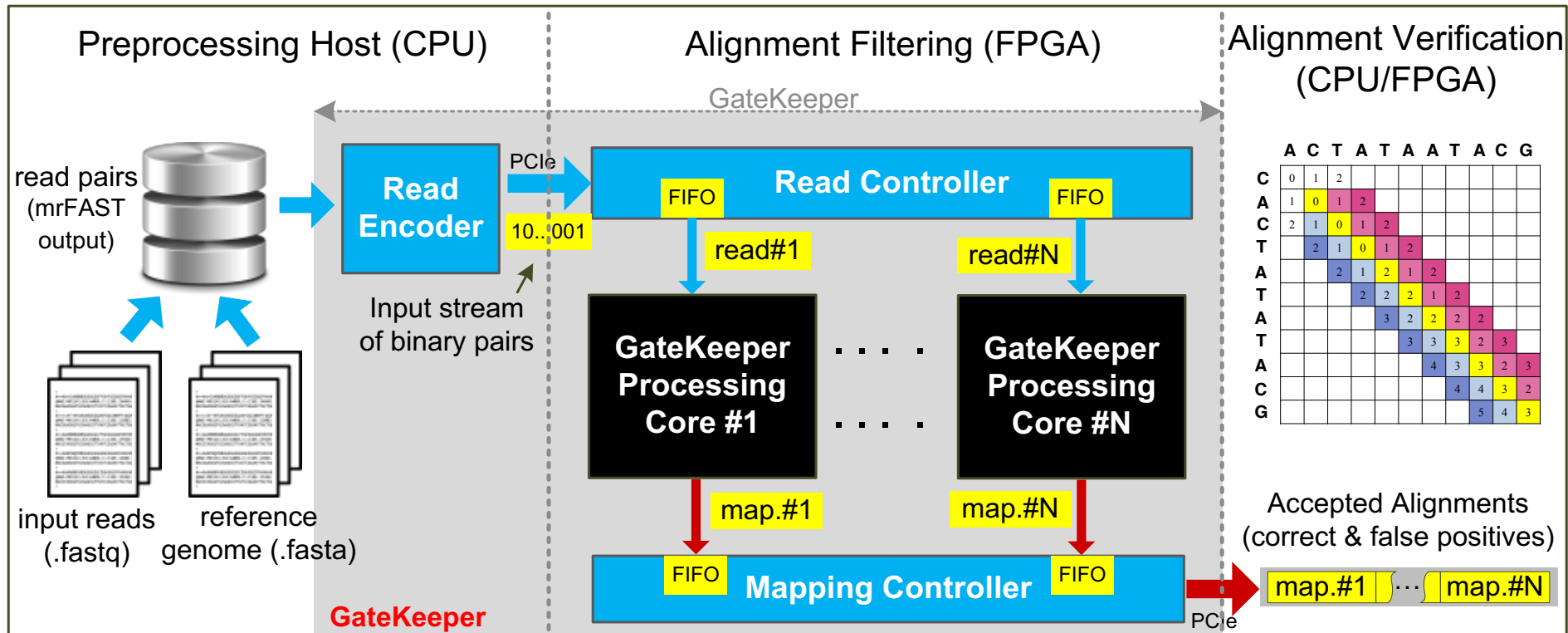
- $(2E) * (\text{ReadLength})$ 2-AND operations.
- $(\text{ReadLength}/4)$ 5-input LUT.
- $\log_2 \text{ReadLength}$ -bit counter.



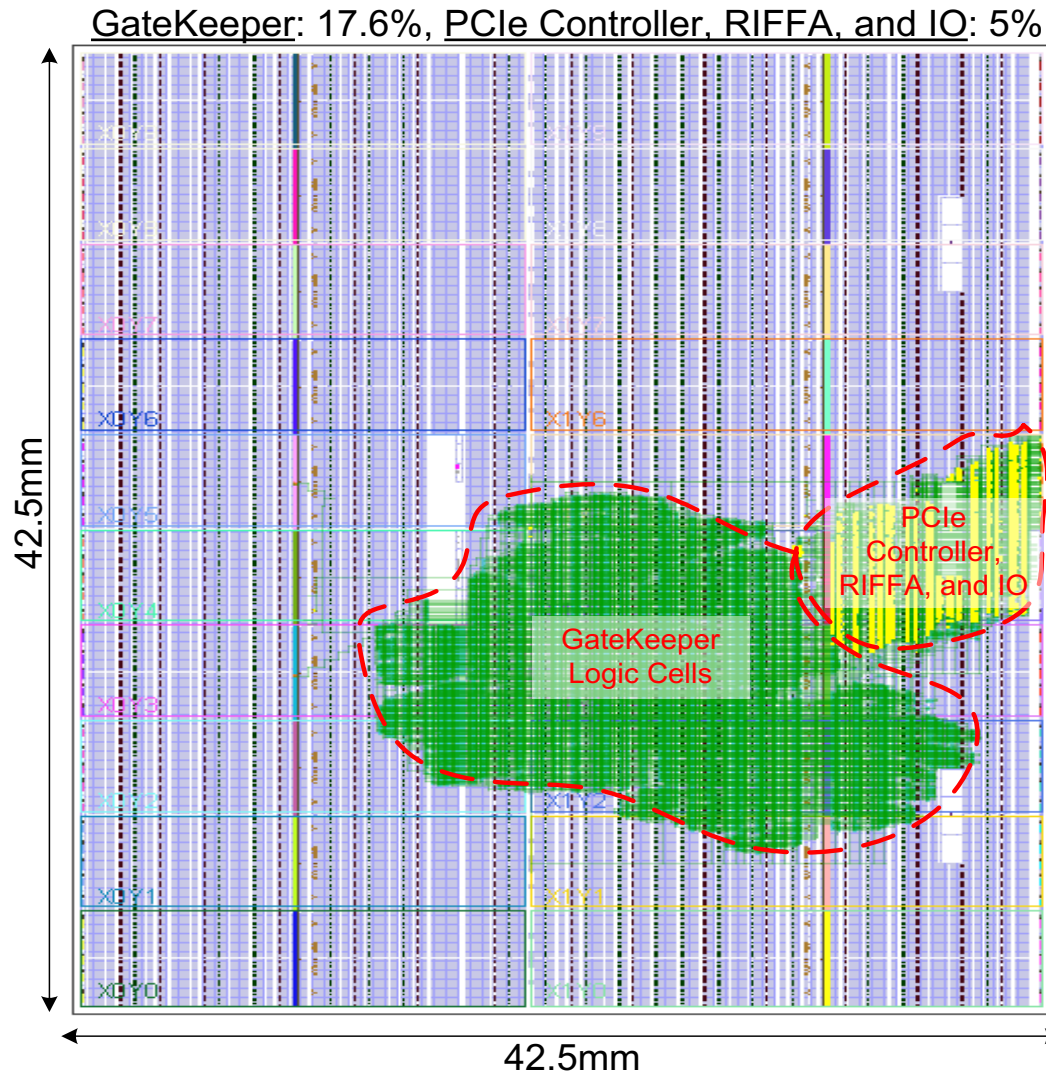
- $(2E+1) * (\text{ReadLength})$ 5-input LUT.

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



300 bp

E=15

GateKeeper: Speed & Accuracy Results

90x-130x faster filter

than SHD (Xin et al., 2015) and the Adjacency Filter (Xin et al., 2013)

4x lower false accept rate

than the Adjacency Filter (Xin et al., 2013)

10x speedup in read mapping

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

Freely available online

github.com/BilkentCompGen/GateKeeper

GateKeeper Conclusions

- **FPGA-based** pre-alignment **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>

Bioinformatics, 31(10), 2015, 1553–1560

doi: 10.1093/bioinformatics/btu856

Advance Access Publication Date: 10 January 2015

Original Paper

OXFORD

Sequence analysis

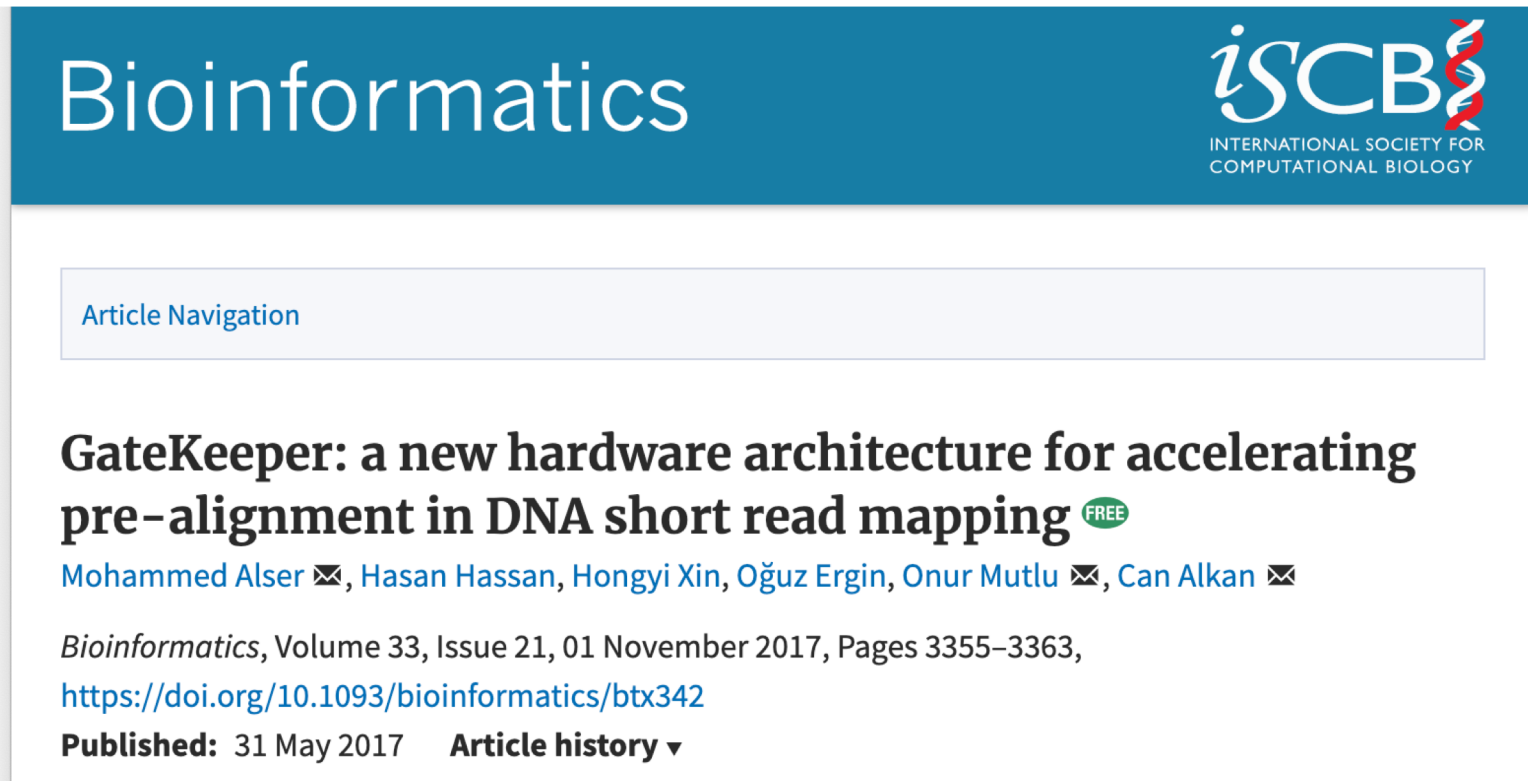
Shifted Hamming distance: a fast and accurate SIMD-friendly filter to accelerate alignment verification in read mapping

**Hongyi Xin^{1,*}, John Greth², John Emmons², Gennady Pekhimenko¹,
Carl Kingsford³, Can Alkan^{4,*} and Onur Mutlu^{2,*}**

More on GateKeeper

- Download and test for yourself

<https://github.com/BilkentCompGen/GateKeeper>



The screenshot shows the top section of a Bioinformatics article page. At the top, there is a blue header bar with the word "Bioinformatics" in white on the left and the "iSCB" logo (International Society for Computational Biology) on the right. Below the header, there is a light blue box labeled "Article Navigation". The main title of the article is "GateKeeper: a new hardware architecture for accelerating pre-alignment in DNA short read mapping", with a green "FREE" badge next to it. Below the title, the authors are listed: "Mohammed Alser", "Hasan Hassan", "Hongyi Xin", "Oğuz Ergin", "Onur Mutlu", and "Can Alkan", each followed by an email icon. Below the authors, the journal information is provided: "Bioinformatics, Volume 33, Issue 21, 01 November 2017, Pages 3355–3363," followed by the DOI link "https://doi.org/10.1093/bioinformatics/btx342". At the bottom of the article preview, it says "Published: 31 May 2017" and "Article history" with a dropdown arrow.

Bioinformatics

iSCB
INTERNATIONAL SOCIETY FOR
COMPUTATIONAL BIOLOGY

Article Navigation

GateKeeper: a new hardware architecture for accelerating pre-alignment in DNA short read mapping FREE

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

Bioinformatics, Volume 33, Issue 21, 01 November 2017, Pages 3355–3363,
<https://doi.org/10.1093/bioinformatics/btx342>

Published: 31 May 2017 **Article history** ▼

Alser+, "[GateKeeper: A New Hardware Architecture for Accelerating Pre-Alignment in DNA Short Read Mapping](#)", *Bioinformatics*, 2017.

Can we do better? Scalability?

Sequence alignment

Shouji: a fast and efficient pre-alignment filter for sequence alignment

Mohammed Alser^{1,2,3,*}, Hasan Hassan¹, Akash Kumar², Onur Mutlu^{1,3,*} and Can Alkan^{3,*}

¹Computer Science Department, ETH Zürich, Zürich 8092, Switzerland, ²Chair for Processor Design, Center For Advancing Electronics Dresden, Institute of Computer Engineering, Technische Universität Dresden, 01062 Dresden, Germany and ³Computer Engineering Department, Bilkent University, 06800 Ankara, Turkey

*To whom correspondence should be addressed.

Associate Editor: Inanc Birol

Received on September 13, 2018; revised on February 27, 2019; editorial decision on March 7, 2019; accepted on March 27, 2019

Alser+, "[Shouji: a fast and efficient pre-alignment filter for sequence alignment](https://doi.org/10.1093/bioinformatics/btz234)", *Bioinformatics* 2019, <https://doi.org/10.1093/bioinformatics/btz234>

■ **Key observation:**

- ❑ Correct alignment always includes **long identical subsequences**.
- ❑ Processing the entire mapping at once is ineffective for hardware design.

■ **Key idea:**

- ❑ Use **overlapping sliding window** approach to quickly and accurately find all long segments of **consecutive zeros**.

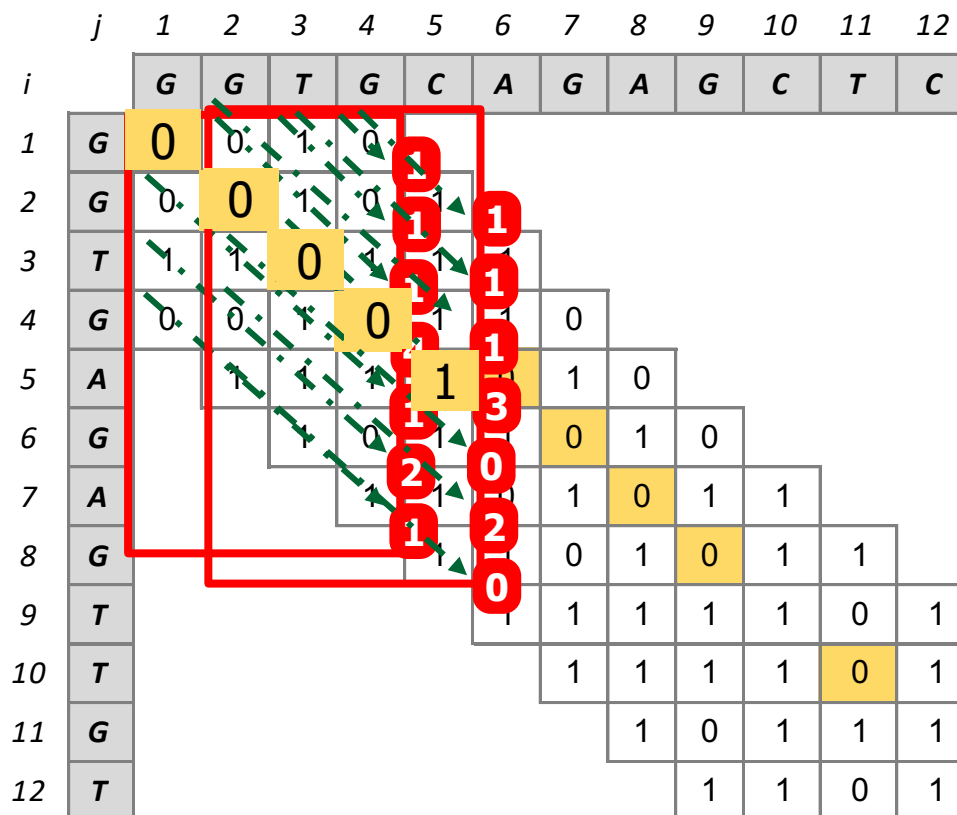
■ **Key result:**

- ❑ Shouji on FPGA is at least **160x faster** than its CPU implementation.
- ❑ Shouji accelerates **best-performing CPU read aligner Edlib** (Bioinformatics 2017) by **up to 18.8x** using 16 filtering units that work in parallel.
- ❑ Shouji is **2.4x to 467x more accurate** than GateKeeper (Bioinformatics 2017) and SHD (Bioinformatics 2015).

Shouji Walkthrough

Building the
Neighborhood Map

Finding all common
subsequences
(diagonal segments of
consecutive zeros)
shared between two
given sequences.



Storing it @ Shouji Bit-vector

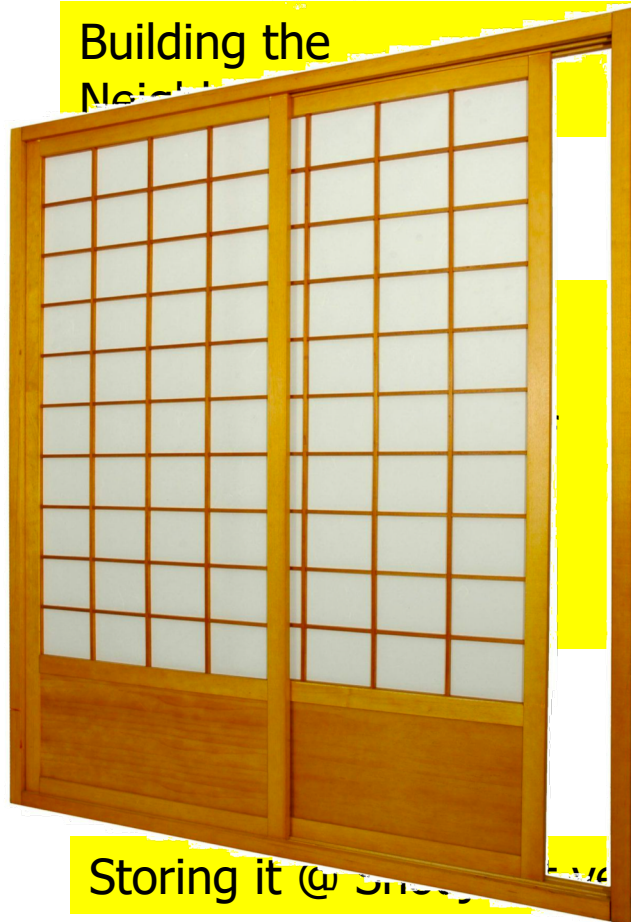
0	0	0	0	1	0	0	0	0	1	0	1
---	---	---	---	---	---	---	---	---	---	---	---

ACCEPT iff number of '1' \leq Threshold

Shouji: a fast and efficient pre-alignment filter for sequence alignment, *Bioinformatics* 2019,
<https://doi.org/10.1093/bioinformatics/btz234>

Shouji Walkthrough

Building the
Neighborhood



Storing it @ Shouji Vector

<i>j</i>	1	2	3	4	5	6	7	8	9	10	11	12	
<i>i</i>	G	G	T	G	C	A	G	A	G	C	T	C	
1	G	0	0	1	0								
2	G	0	0	1	0	1							
3	T	1	1	0	1	1	1						
4	G	0	0	1	0	1	1	0					
5	A		1	1	1	1	0	1	0				
6	G			1	0	1	1	0	1	0			
7	A				1	1	0	1	0	1	1		
8	G					1	1	0	1	0	1	1	
9	T						1	1	1	1	1	0	1
10	T							1	1	1	1	0	1
11	G								1	0	1	1	1
12	T									1	1	0	1

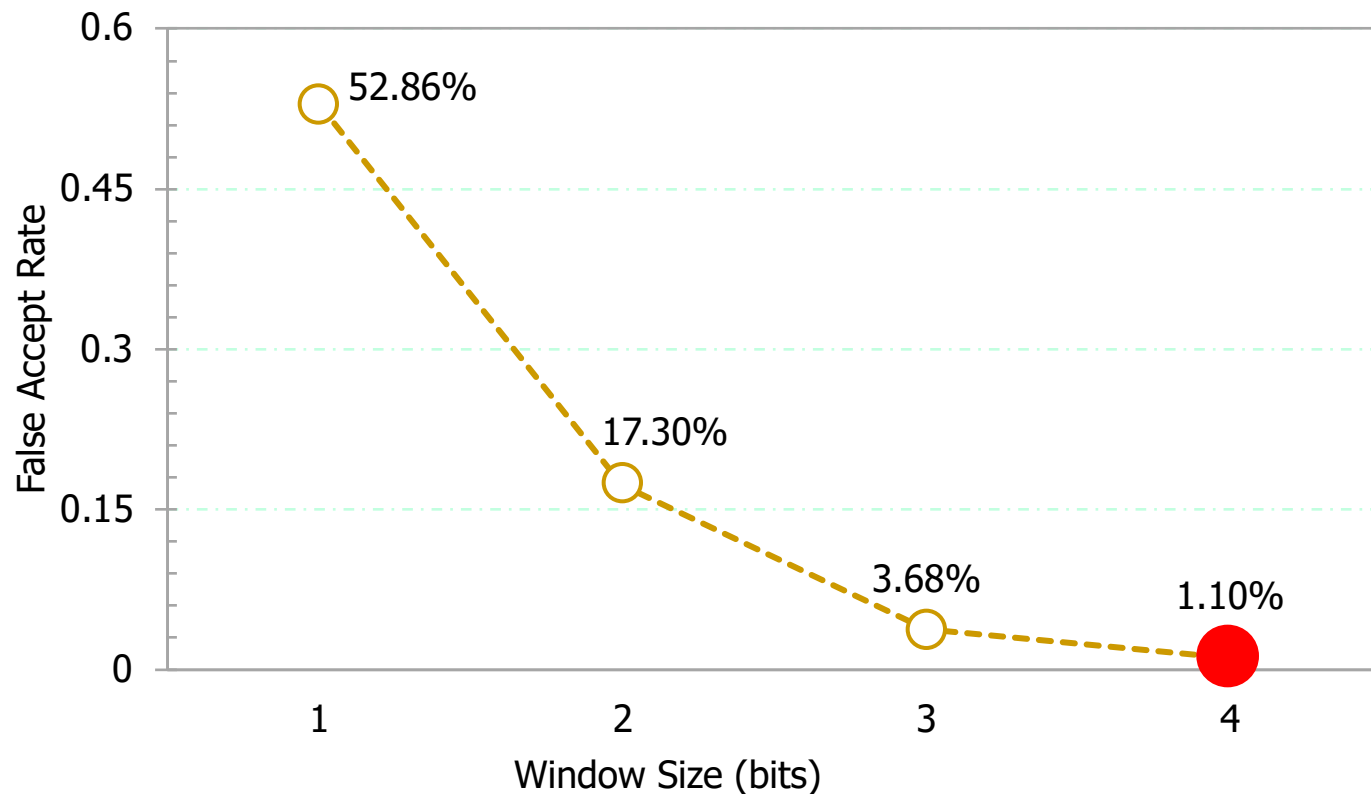
0 0 0 0 1 0 0 0 0 0 1 0 1

ACCEPT iff number of '1' \leq Threshold

Shouji: a fast and efficient pre-alignment filter for sequence alignment, *Bioinformatics* 2019,
<https://doi.org/10.1093/bioinformatics/btz234>

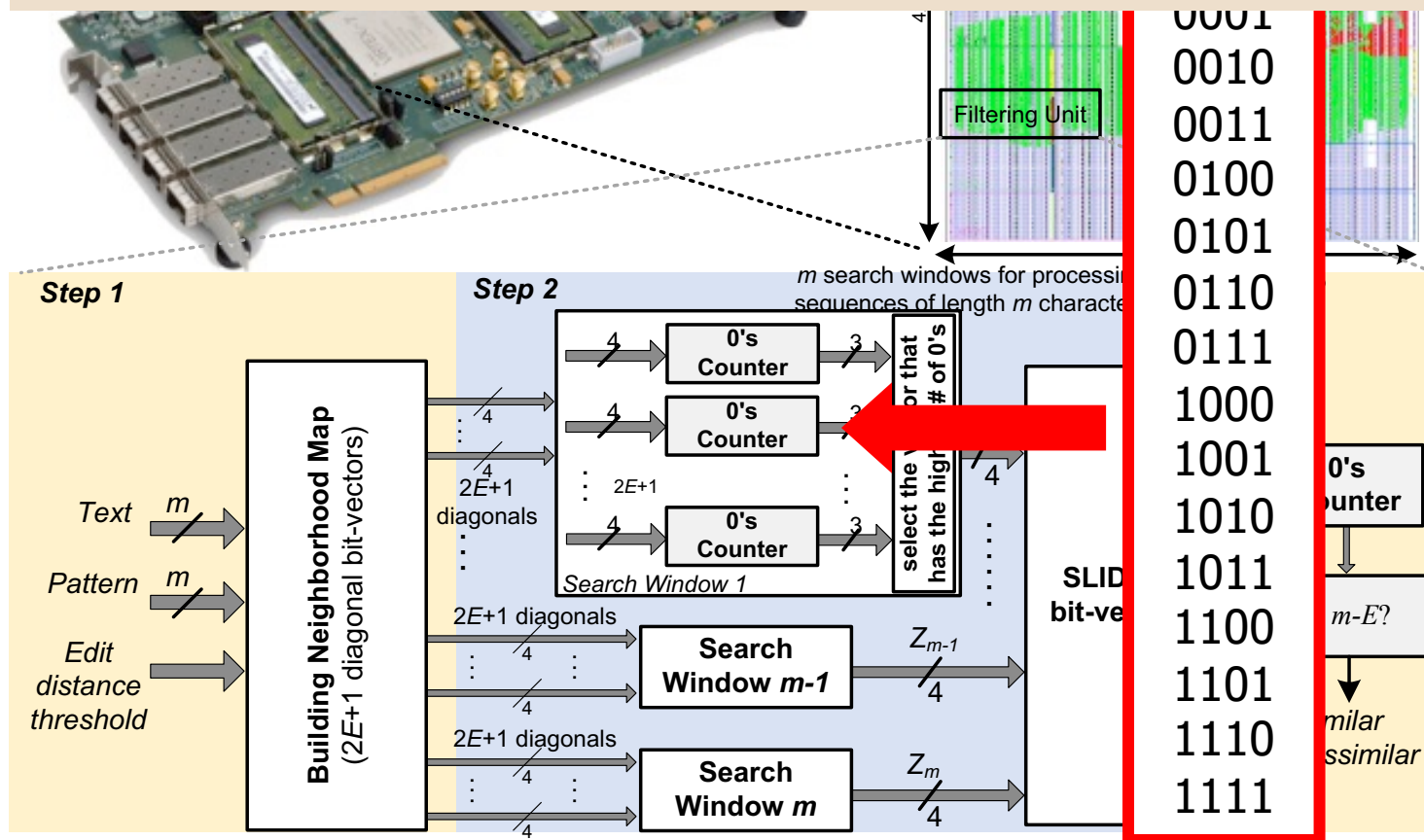
Sliding Window Size

- The reason behind the selection of the window size is due to the minimal possible length of the identical subsequence that is a single match (e.g., such as `101').



Hardware Implementation

- Counting is performed **concurrently** for **all** bit-vectors and all sliding windows in a single clock cycle using **multiple 4-input LUTs**.



More on Shouji

Download and test for yourself

<https://github.com/CMU-SAFARI/Shouji>

Bioinformatics, 2019, 1–9

doi: 10.1093/bioinformatics/btz234

Advance Access Publication Date: 28 March 2019

Original Paper

OXFORD

Sequence alignment

Shouji: a fast and efficient pre-alignment filter for sequence alignment

Mohammed Alser^{1,2,3,*}, Hasan Hassan¹, Akash Kumar², Onur Mutlu^{1,3,*} and Can Alkan^{3,*}

¹Computer Science Department, ETH Zürich, Zürich 8092, Switzerland, ²Chair for Processor Design, Center For Advancing Electronics Dresden, Institute of Computer Engineering, Technische Universität Dresden, 01062 Dresden, Germany and ³Computer Engineering Department, Bilkent University, 06800 Ankara, Turkey

*To whom correspondence should be addressed.

Associate Editor: Inanc Birol

Received on September 13, 2018; revised on February 27, 2019; editorial decision on March 7, 2019; accepted on March 27, 2019

Alser+, "[Shouji: a fast and efficient pre-alignment filter for sequence alignment](https://doi.org/10.1093/bioinformatics/btz234)", *Bioinformatics* 2019, <https://doi.org/10.1093/bioinformatics/btz234>

SneakySnake

SneakySnake: A Fast and Accurate Universal Genome Pre-Alignment Filter for CPUs, GPUs, and FPGAs

Mohammed Alser^{1,3}, Taha Shahroodi¹, Juan Gómez-Luna¹, Can Alkan³, and Onur Mutlu^{1,2,3}

¹Department of Computer Science, ETH Zurich, Zurich 8006, Switzerland

²Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh 15213, PA, USA

³Department of Computer Engineering, Bilkent University, Ankara 06800, Turkey

Alser + "[SneakySnake: A Fast and Accurate Universal Genome Pre-Alignment Filter for CPUs, GPUs, and FPGAs.](#)" *arXiv preprint* (2019).

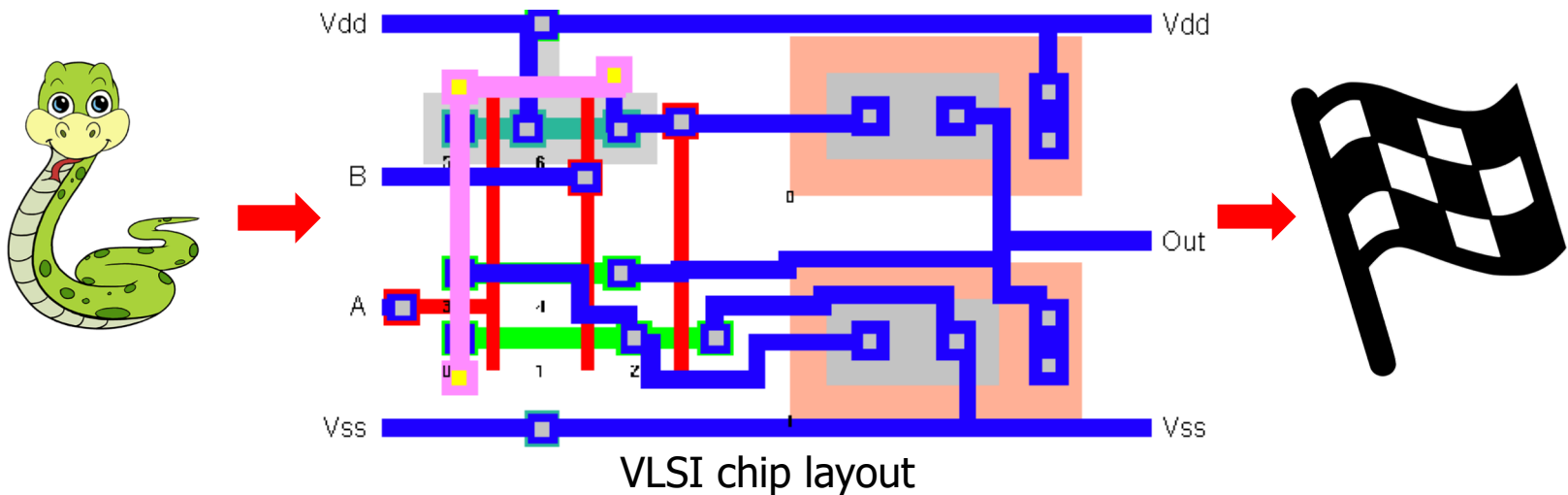
SneakySnake

■ 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.



SneakySnake

■ 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.

■ Key result:

- SneakySnake is up to four orders of magnitude more accurate than Shouji (Bioinformatics'19) and GateKeeper (Bioinformatics'17).
- SneakySnake accelerates the state-of-the-art CPU-based sequence aligners, Edlib (Bioinformatics'17) and Parasail (BMC Bioinformatics'16), by up to 37.6× and 43.9× (>12× on average), respectively, *without requiring hardware acceleration*, and by up to 413× and 689× (>400× on average), respectively, *using hardware acceleration*.

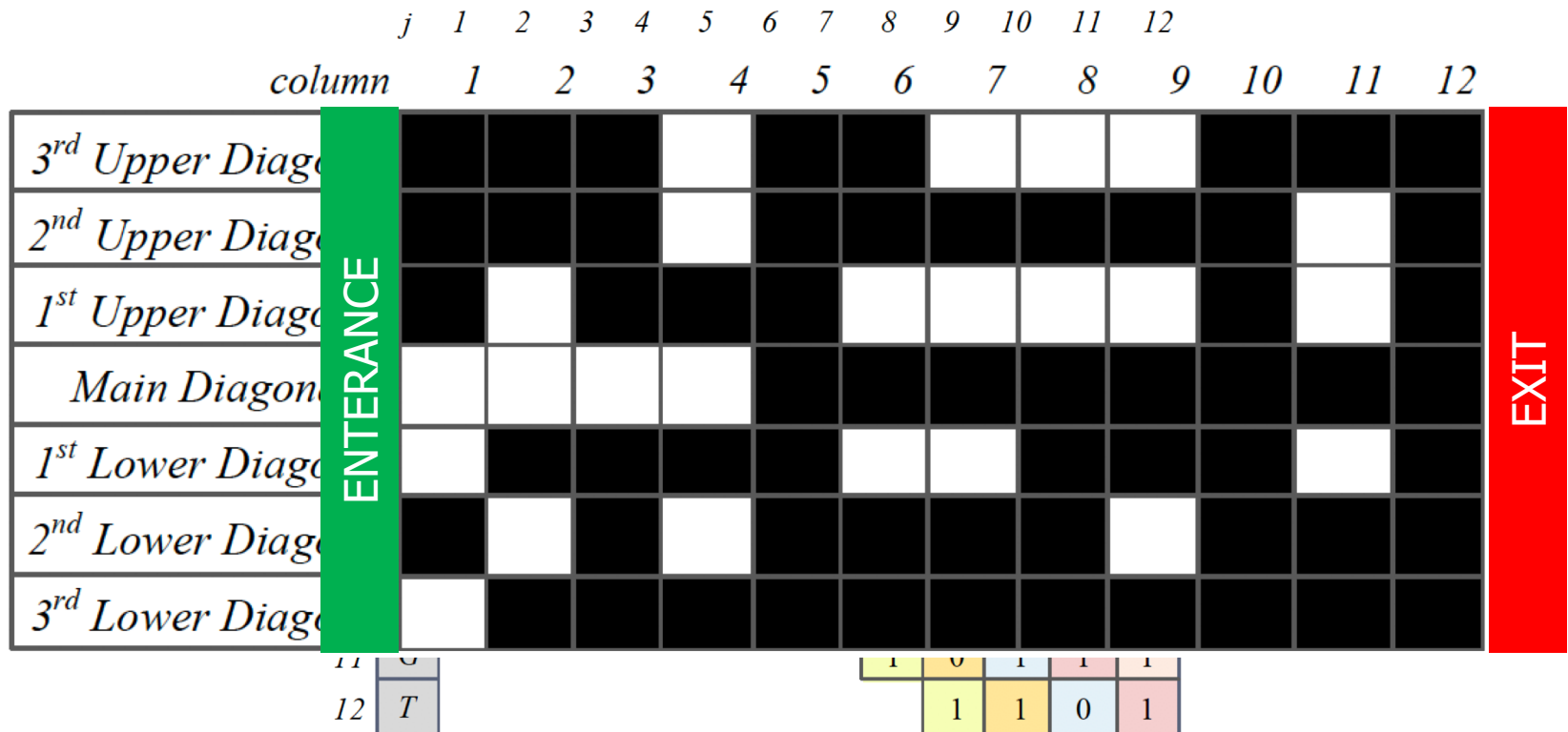
SneakySnake Walkthrough

Building Neighborhood Map

Finding the Optimal Routing Path

Examining the Snake Survival

$$E = 3$$

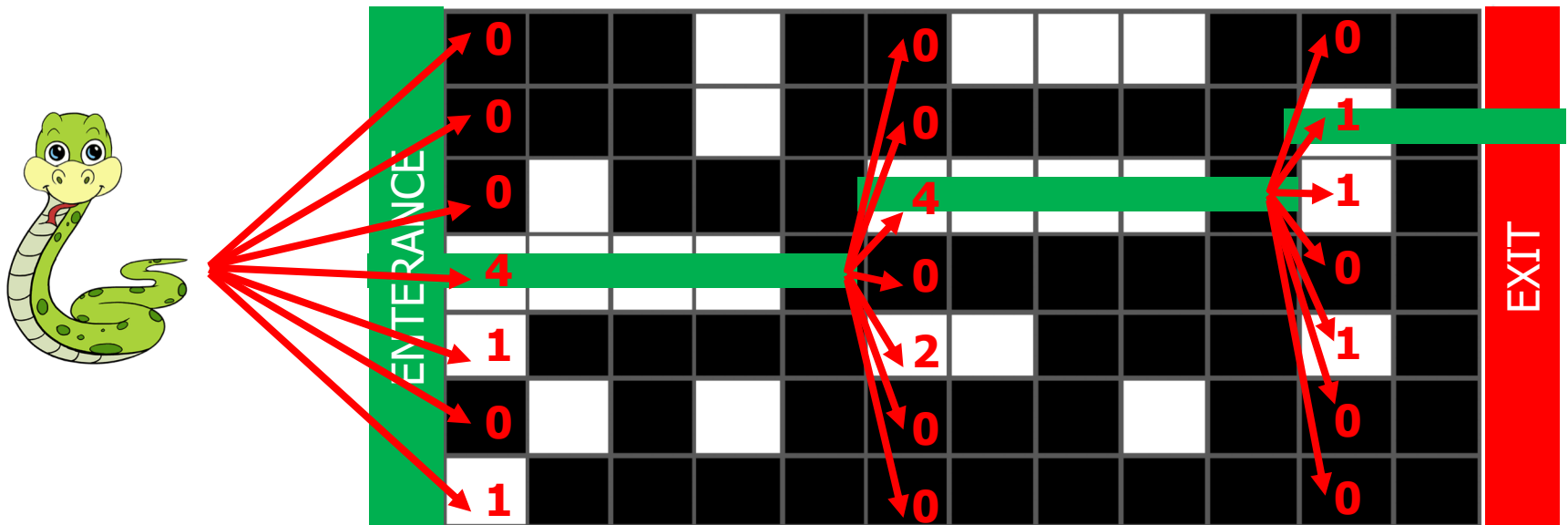


SneakySnake Walkthrough

Building Neighborhood Map

Finding the Optimal Routing Path

Examining the Snake Survival



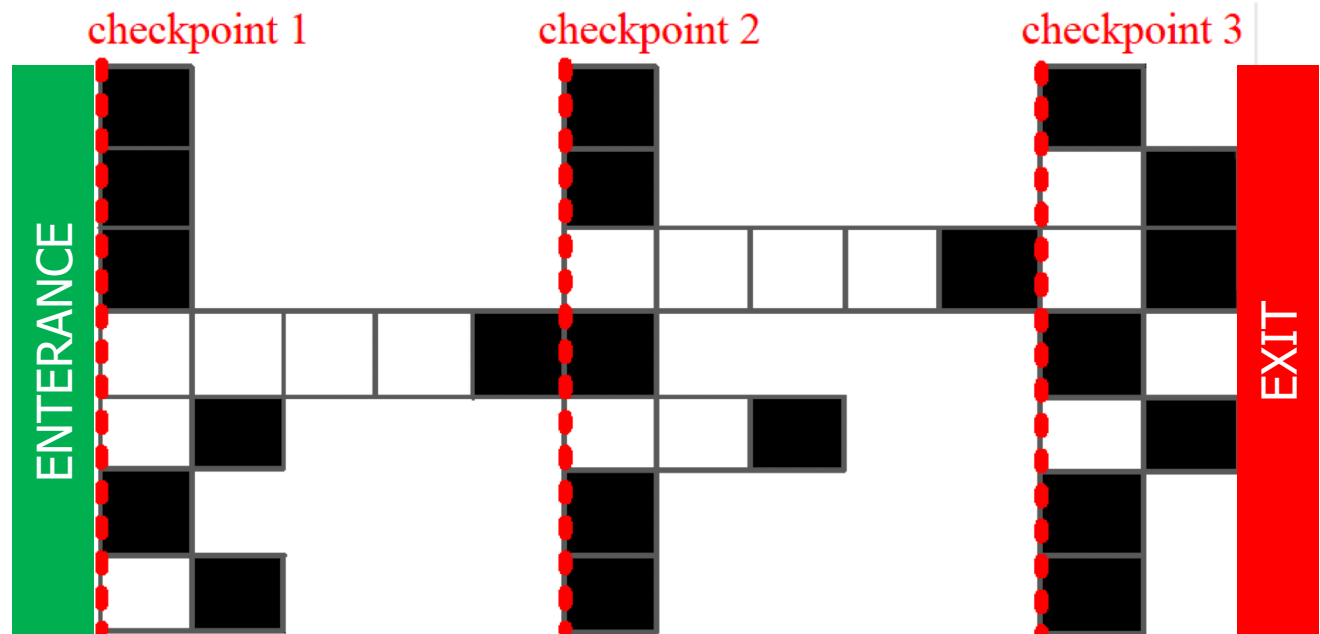
SneakySnake Walkthrough

Building Neighborhood Map

Finding the Routing Travel Path

Examining the Snake Survival

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).

	E (bp)	Slice LUT	Slice Register	No. of Filtering Units
GateKeeper	2	0.39%	0.01%	16
	5	0.71%	0.01%	16
Shouji	2	0.69%	0.08%	16
	5	1.72%	0.16%	16
Snake-on-Chip	2	0.68%	0.16%	16
	5	1.42%	0.34%	16

SneakySnake

SneakySnake: A Fast and Accurate Universal Genome Pre-Alignment Filter for CPUs, GPUs, and FPGAs

Mohammed Alser^{1,3}, Taha Shahroodi¹, Juan Gómez-Luna¹, Can Alkan³, and Onur Mutlu^{1,2,3}

¹Department of Computer Science, ETH Zurich, Zurich 8006, Switzerland

²Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh 15213, PA, USA

³Department of Computer Engineering, Bilkent University, Ankara 06800, Turkey

Download and test for CPU, GPU, and FPGA:

<https://github.com/CMU-SAFARI/SneakySnake>

Alser + "[SneakySnake: A Fast and Accurate Universal Genome Pre-Alignment Filter for CPUs, GPUs, and FPGAs.](#)" *arXiv preprint* (2019).

Read Mapping & Filtering

- Problem: Heavily bottlenecked by Data Movement
- Shouji performance limited by DRAM bandwidth [Alser+, Bioinformatics 2019]
- GateKeeper performance limited by DRAM bandwidth [Alser+, Bioinformatics 2017]
- Ditto for SHD [Xin+, Bioinformatics 2015]
- Solution: Processing-in-memory can alleviate the bottleneck

Read Mapping & Filtering in Memory

We need to design
mapping & filtering algorithms
that fit processing-in-memory

GRIM-Filter

- Jeremie S. Kim, Damla Senol Cali, Hongyi Xin, Donghyuk Lee, Saugata Ghose, Mohammed Alser, Hasan Hassan, Oguz Ergin, Can Alkan, and Onur Mutlu, **"GRIM-Filter: Fast Seed Location Filtering in DNA Read Mapping Using Processing-in-Memory Technologies"** to appear in ***BMC Genomics***, 2018. *Proceedings of the 16th Asia Pacific Bioinformatics Conference (APBC)*, Yokohama, Japan, January 2018. [arxiv.org Version \(pdf\)](#)

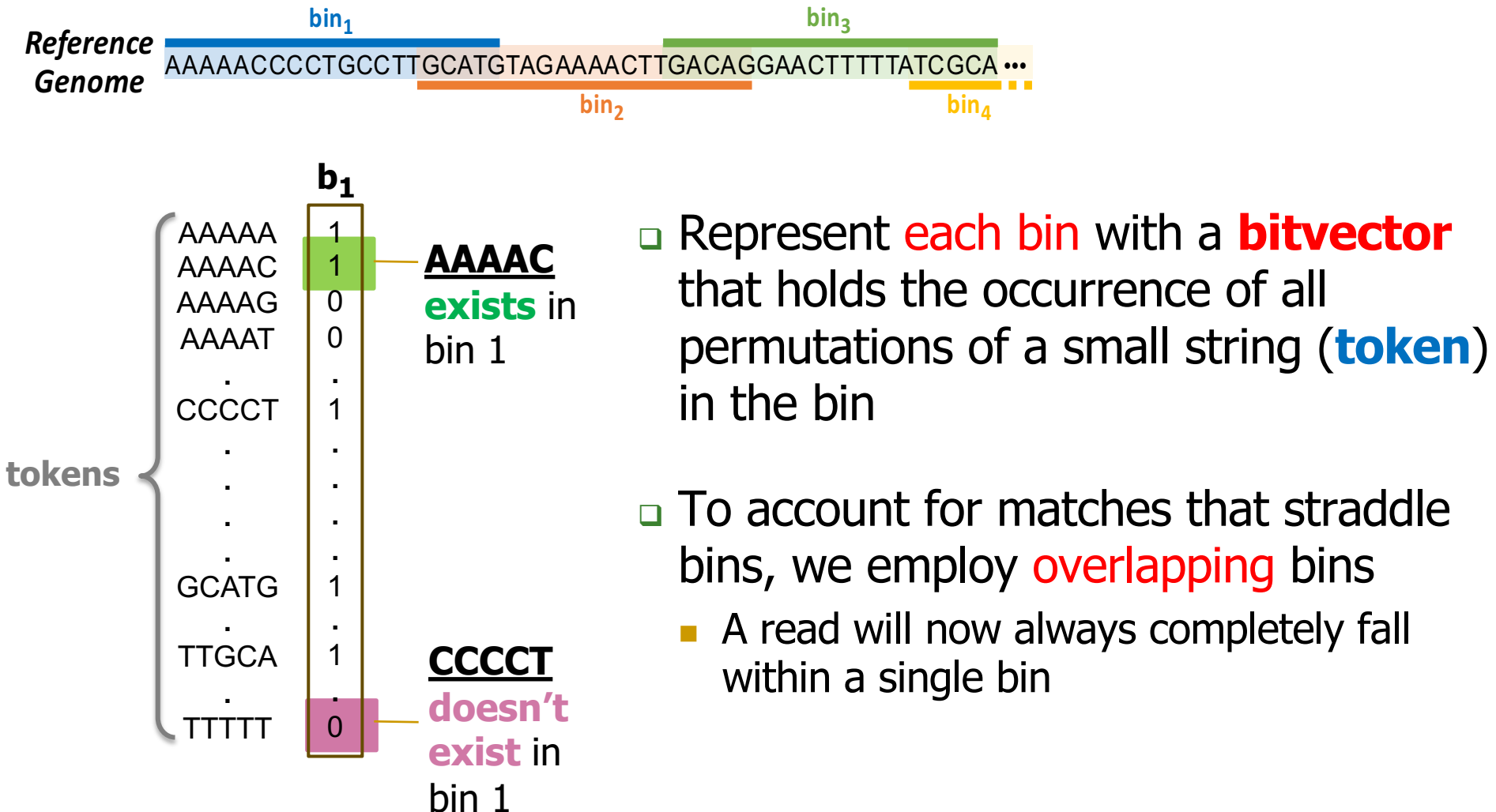
GRIM-Filter: Fast Seed Location Filtering in DNA Read Mapping Using Processing-in-Memory Technologies

Jeremie S. Kim^{1,6*}, Damla Senol Cali¹, Hongyi Xin², Donghyuk Lee³, Saugata Ghose¹, Mohammed Alser⁴, Hasan Hassan⁶, Oguz Ergin⁵, Can Alkan^{*4}, and Onur Mutlu^{*6,1}

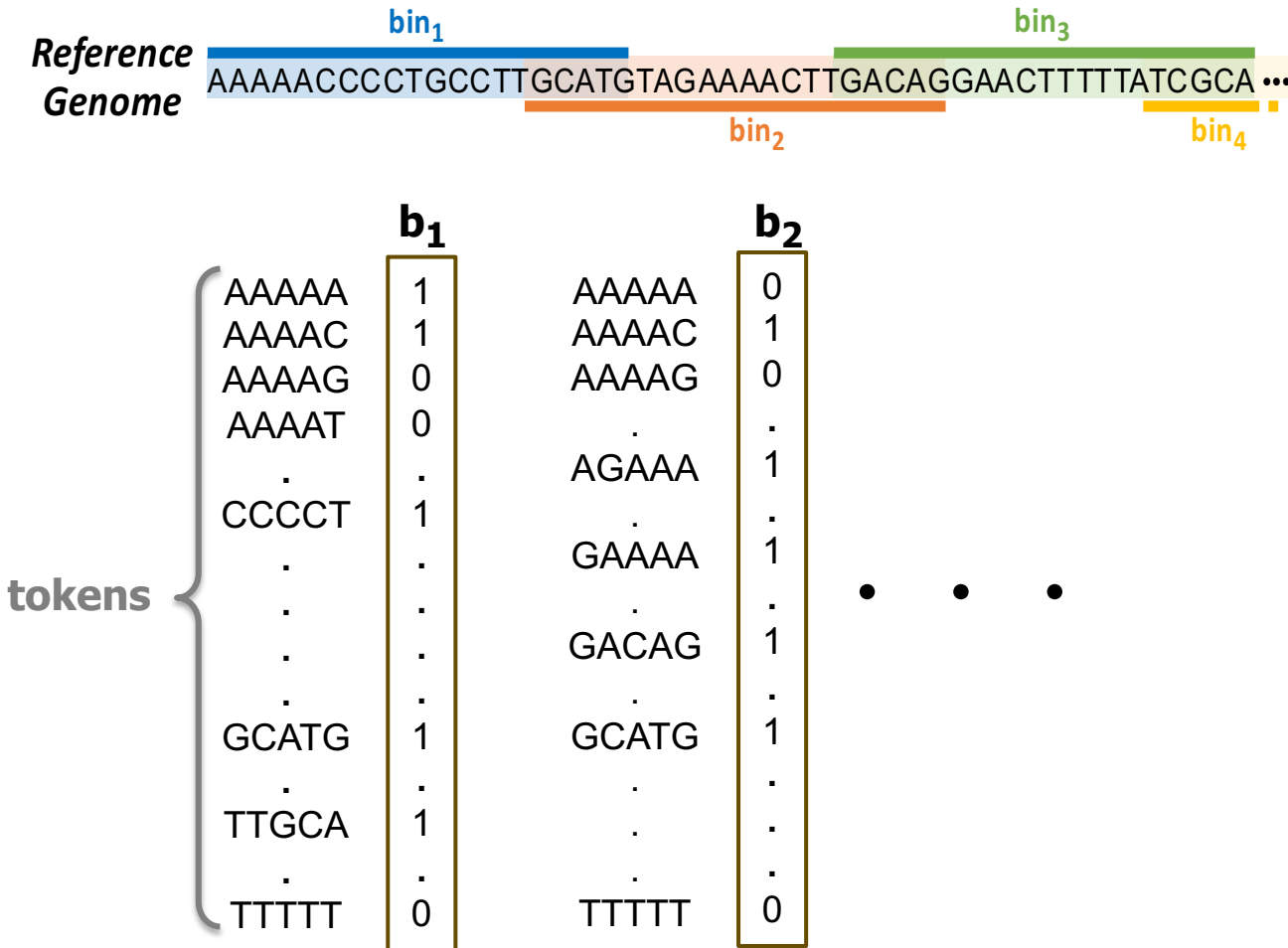
GRIM-Filter

- **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.

GRIM-Filter: Bitvectors



GRIM-Filter: Bitvectors

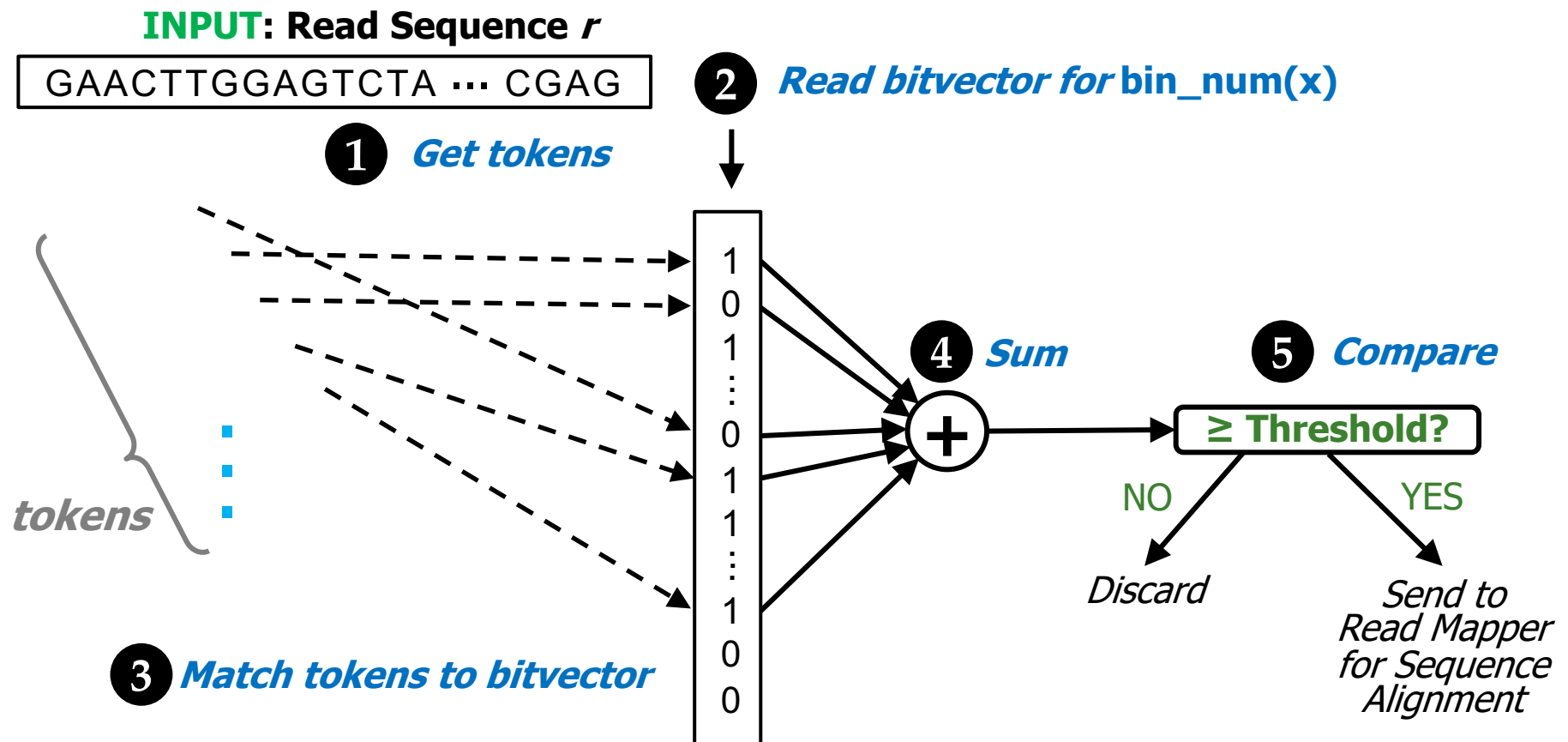


Storing all bitvectors requires $4^n * t$ bits in memory, where
 t = number of bins
 &
 n = token length.

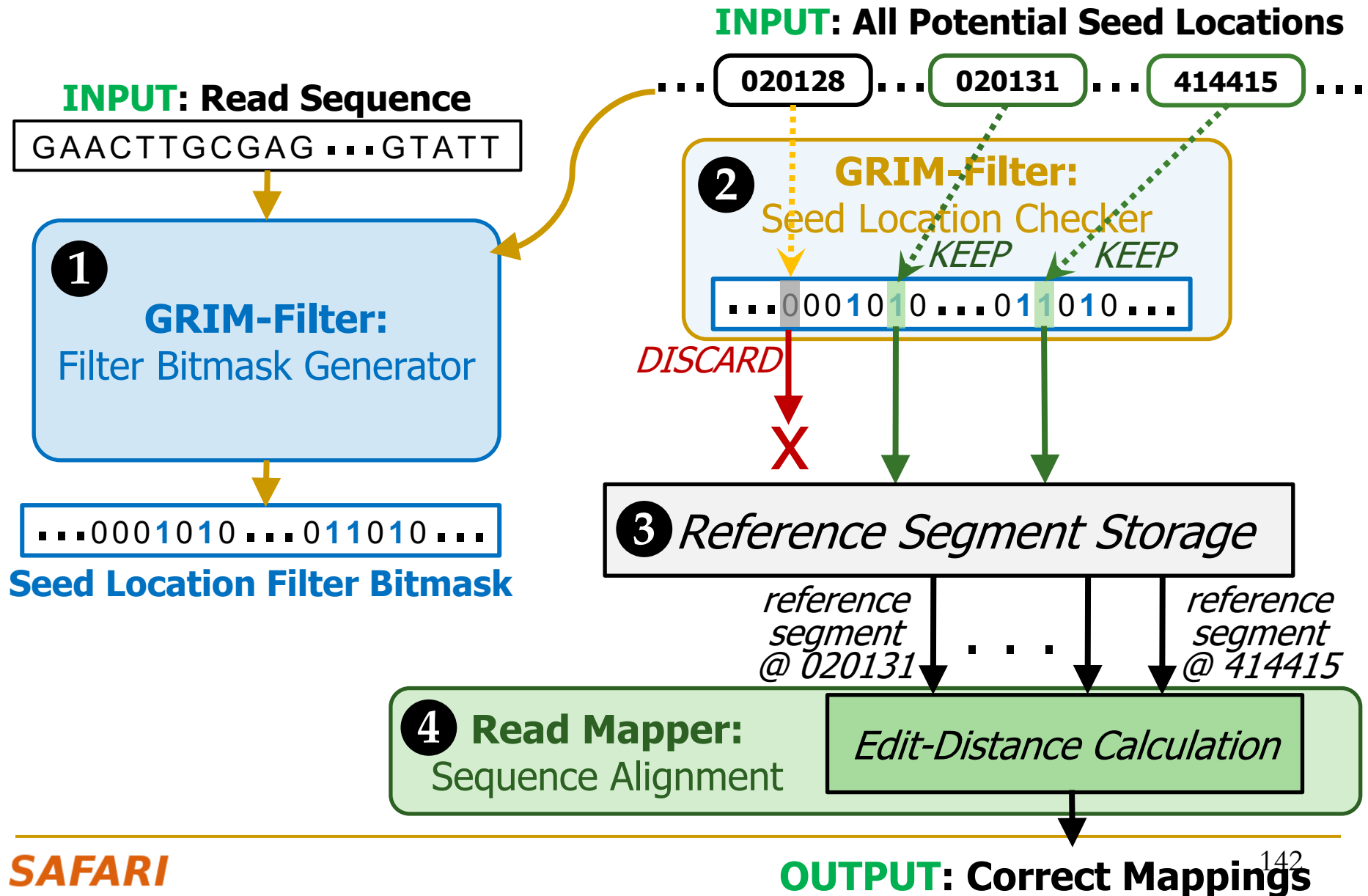
For **bin size** ~200,
 and **n** = 5,
memory footprint
 ~3.8 GB

GRIM-Filter: Checking a Bin

How GRIM-Filter determines whether to **discard** potential match locations in a given bin **prior** to alignment



Integrating GRIM-Filter into a Read Mapper



Key Properties of GRIM-Filter

1. Simple Operations:

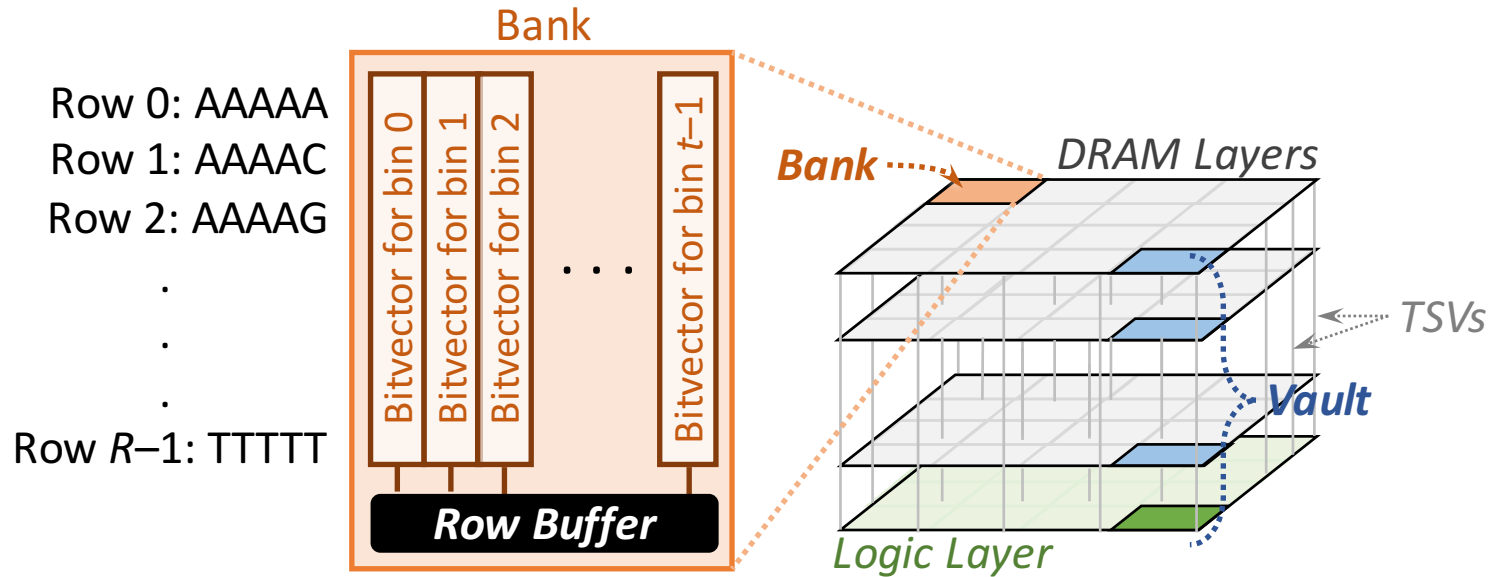
- ❑ To check a given bin, find the **sum** of all bits corresponding to each token in the read
- ❑ **Compare** against threshold to determine whether to align

2. Highly Parallel: Each bin is operated on independently and there are many many bins

3. Memory Bound: Given the frequent accesses to the large bitvectors, we find that GRIM-Filter is memory bound

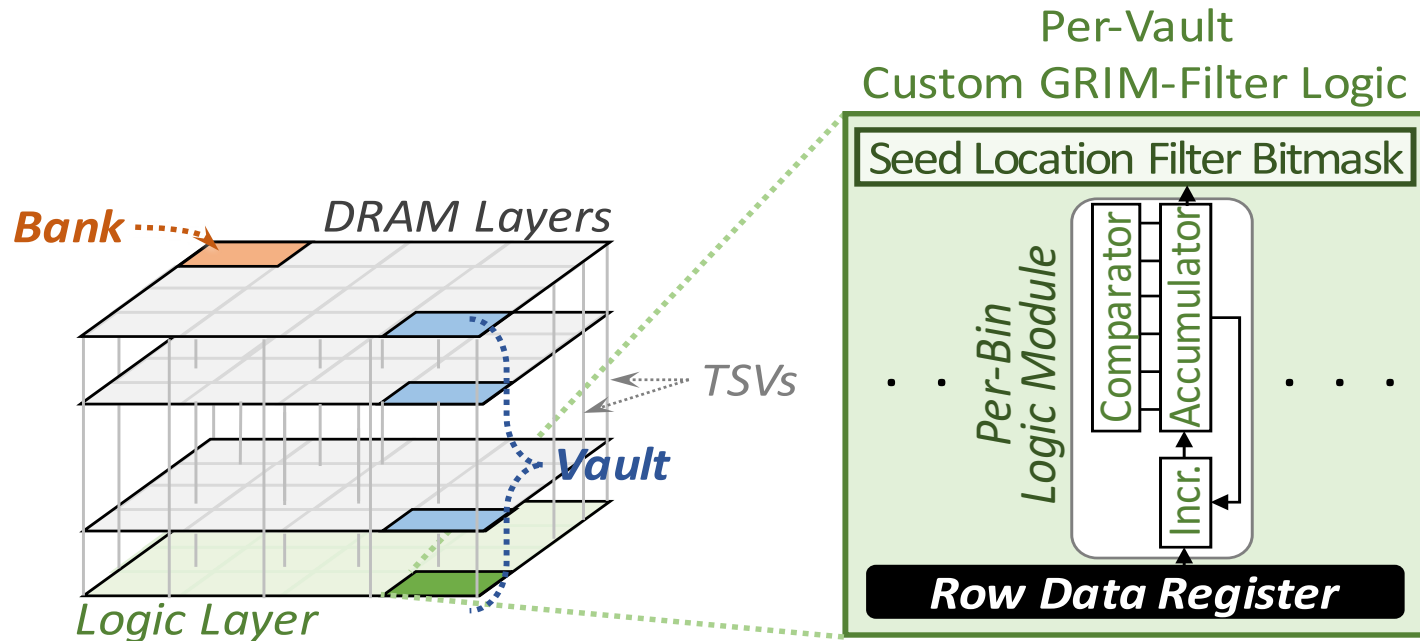
These properties together make GRIM-Filter a good algorithm to be run in 3D-Stacked DRAM

GRIM-Filter in 3D-Stacked DRAM



- 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

GRIM-Filter in 3D-Stacked DRAM



- Customized logic for accumulation and comparison per genome segment
 - Low area overhead, simple implementation
 - For HBM2, we use 4096 incrementer LUTs, 7-bit counters, and comparators in logic layer

More on GRIM-Filter

- Jeremie S. Kim, Damla Senol Cali, Hongyi Xin, Donghyuk Lee, Saugata Ghose, Mohammed Alser, Hasan Hassan, Oguz Ergin, Can Alkan, and Onur Mutlu, **"GRIM-Filter: Fast Seed Location Filtering in DNA Read Mapping Using Processing-in-Memory Technologies"** to appear in ***BMC Genomics***, 2018.
Proceedings of the 16th Asia Pacific Bioinformatics Conference (APBC), Yokohama, Japan, January 2018.
[arxiv.org Version \(pdf\)](#)

GRIM-Filter: Fast Seed Location Filtering in DNA Read Mapping Using Processing-in-Memory Technologies

Jeremie S. Kim^{1,6*}, Damla Senol Cali¹, Hongyi Xin², Donghyuk Lee³, Saugata Ghose¹, Mohammed Alser⁴, Hasan Hassan⁶, Oguz Ergin⁵, Can Alkan^{*4}, and Onur Mutlu^{*6,1}

GenCache: Leveraging In-Cache Operators for Efficient Sequence Alignment

Anirban Nag
anirban@cs.utah.edu
University of Utah
Salt Lake City, Utah

C. N. Ramachandra
ramgowda@cs.utah.edu
University of Utah
Salt Lake City, Utah

Rajeev Balasubramonian
rajeev@cs.utah.edu
University of Utah
Salt Lake City, Utah

Ryan Stutsman
stutsman@cs.utah.edu
University of Utah
Salt Lake City, Utah

Edouard Giacomin
edouard.giacomin@utah.edu
University of Utah
Salt Lake City, Utah

Hari Kambalasubramanyam
hari.kambalasubramanyam@utah.edu
University of Utah
Salt Lake City, Utah

Pierre-Emmanuel Gaillardon
pierre-
emmanuel.gaillardon@utah.edu
University of Utah
Salt Lake City, Utah

Nag, Anirban, et al. "**GenCache: Leveraging In-Cache Operators for Efficient Sequence Alignment**." *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 52)*, ACM, 2019.

GenCache

- **Key observation:** State-of-the-art alignment accelerators are still **bottlenecked by memory**.
- **Key ideas:**
 - ❑ Performing **in-cache alignment + pre-alignment filtering** by enabling processing-in-cache using previous proposal, ComputeCache (HPCA'17).
 - ❑ Using **different Pre-alignment filters** depending on the selected edit distance threshold.
- **Results:**
 - ❑ GenCache on CPU is 1.36x faster than GenAx (ISCA 2018).
GenCache in cache is 5.26x faster than GenAx.
 - ❑ GenCache chip has 16.4% higher area, 34.7% higher peak power, and 15% higher average power than GenAx.

GenCache's Four Phases

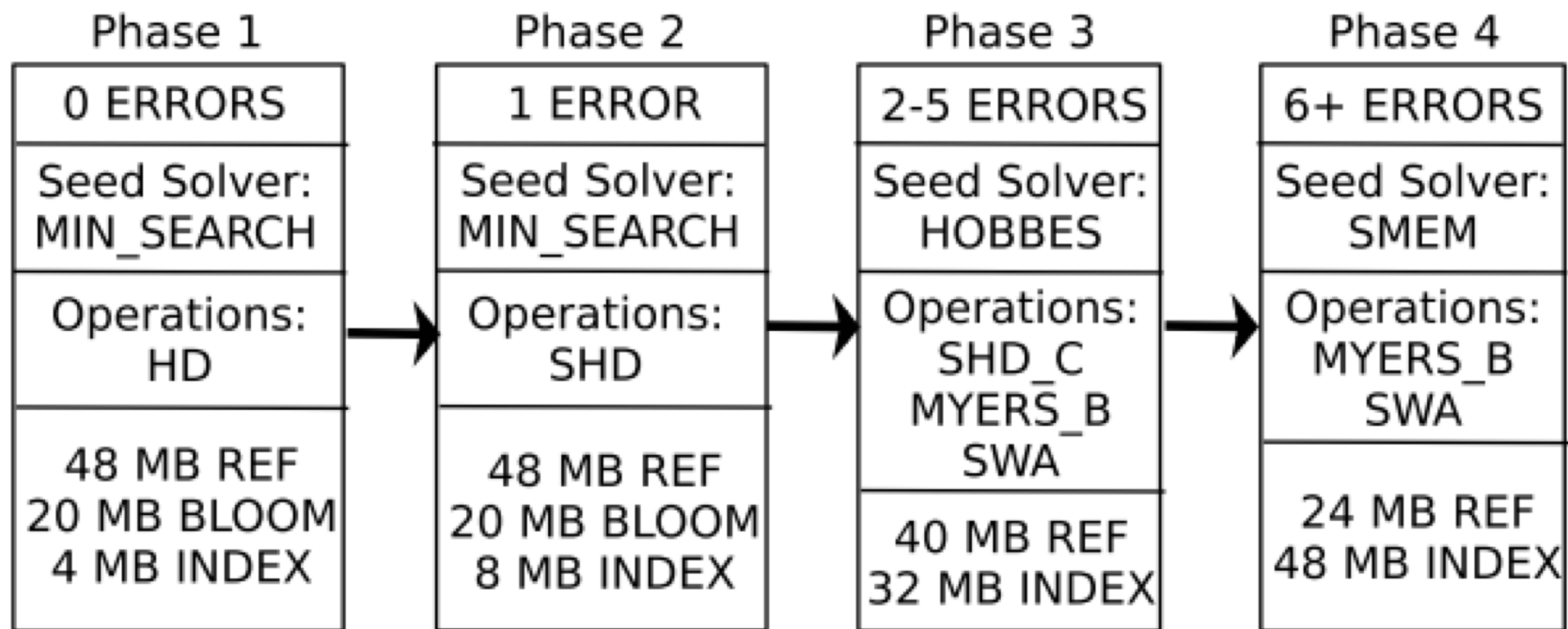


Figure 7: Four phases in the new alignment algorithm that exploits in-cache operators.

Throughput Results

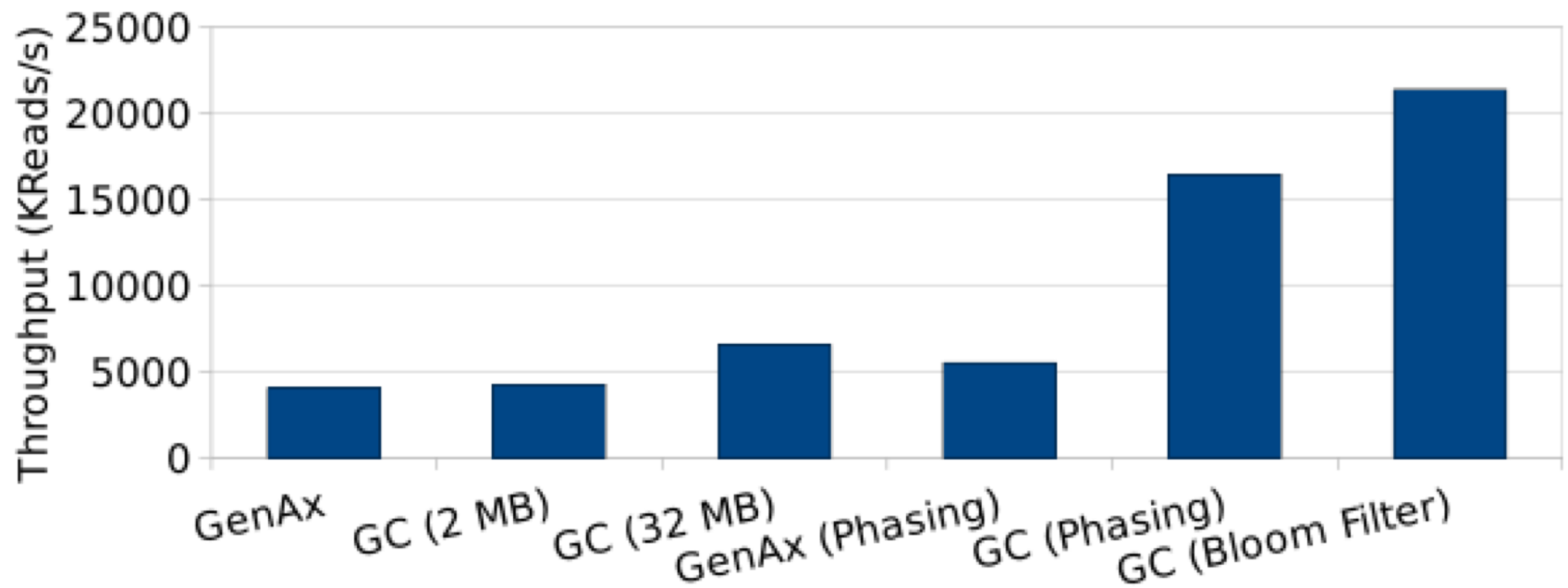


Figure 9: Throughput improvement of GenCache (Hardware & Software).

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.

Darwin: A Genomics Co-processor Provides up to 15,000× acceleration on long read assembly

Yatish Turakhia
Stanford University
yatisht@stanford.edu

Gill Bejerano
Stanford University
bejerano@stanford.edu

William J. Dally
Stanford University
NVIDIA Research
dally@stanford.edu

- Seed filter: **D-Soft**
- Read alignment accelerator: **GACT** ← We will cover this

Yatish+ "Darwin: A genomics co-processor provides up to 15,000 x acceleration on long read assembly." *ASPLOS* 2018.

<http://bejerano.stanford.edu/papers/p199-turakhia.pdf>

Darwin: GACT Hardware Acceleration

■ Key observation:

- ❑ **Data Dependencies limit** accelerating the dynamic programming table calculation.

■ Key idea:

- ❑ **Divide** the dynamic programming table into **overlapping *tiles***.
- ❑ Calculate each tile **independently** and in a **systolic array** fashion.
- ❑ Calculate **many** alignments **concurrently**.

■ Key result:

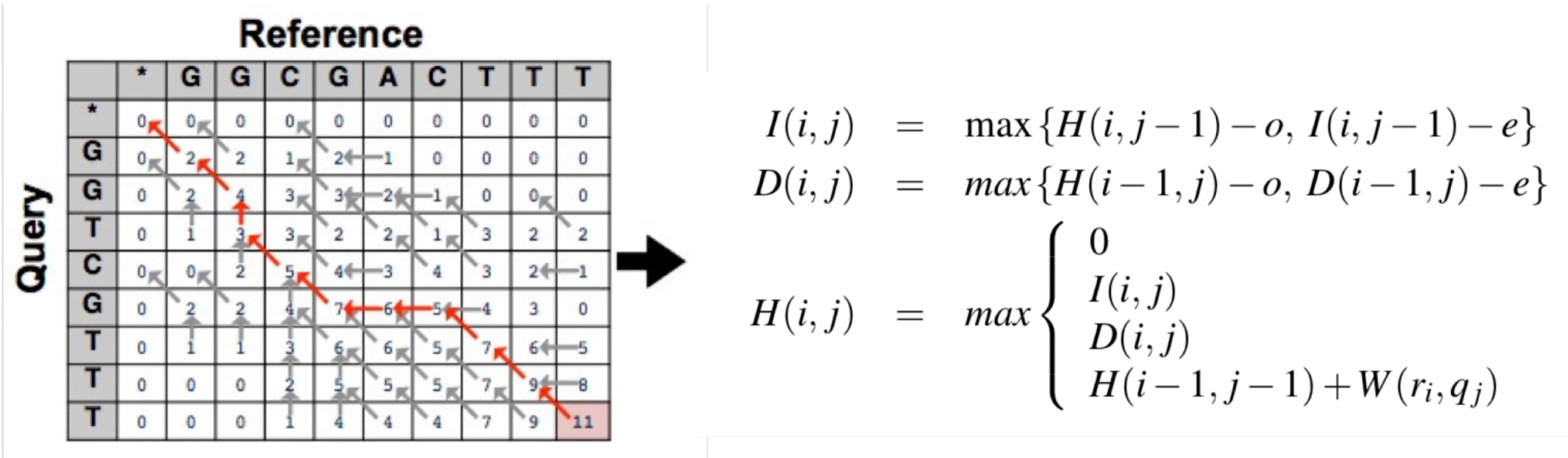
- ❑ It is simulated for TSMC 40nm **CMOS** process.
- ❑ It provides a **speedup** of up to **380x** compared to GACT **software**.
- ❑ It is **three orders** of magnitude **faster** than **Edlib** (best-performing CPU read aligner).

■ Weaknesses:

- ❑ It is not clear if tiling **maintains the same accuracy** as the original dynamic programming algorithm.

Specialized Accelerator for Read Aligner

- Accelerating the read alignment algorithm as-is using specialized hardware (40 nm CMOS) provides a **limited speedup** (37x).



Dynamic programming for gene sequence alignment (Smith-Waterman)

CPU-based read aligner

vs.

Hardware accelerated read aligner

On 14nm CPU

On 40nm Special Unit

35 ALU ops, 15 load/store

1 cycle (37x speedup)

37 cycles

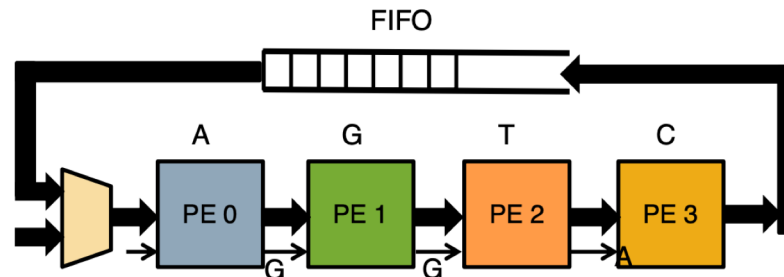
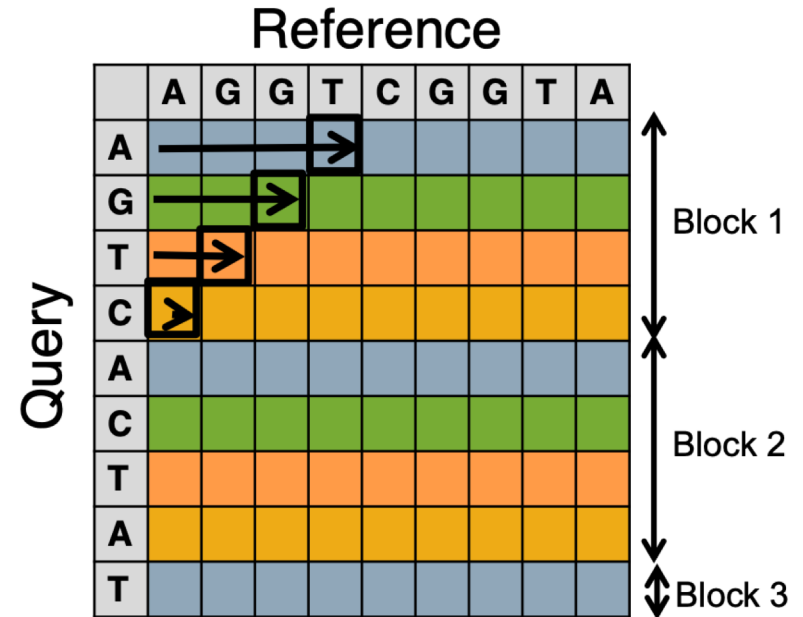
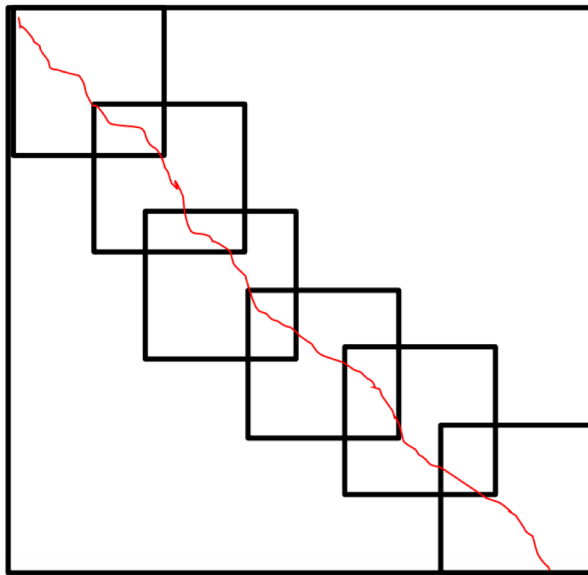
3.1pJ (26,000x efficiency)

81nJ

300fJ for logic (remainder is memory)

GACT Alignment

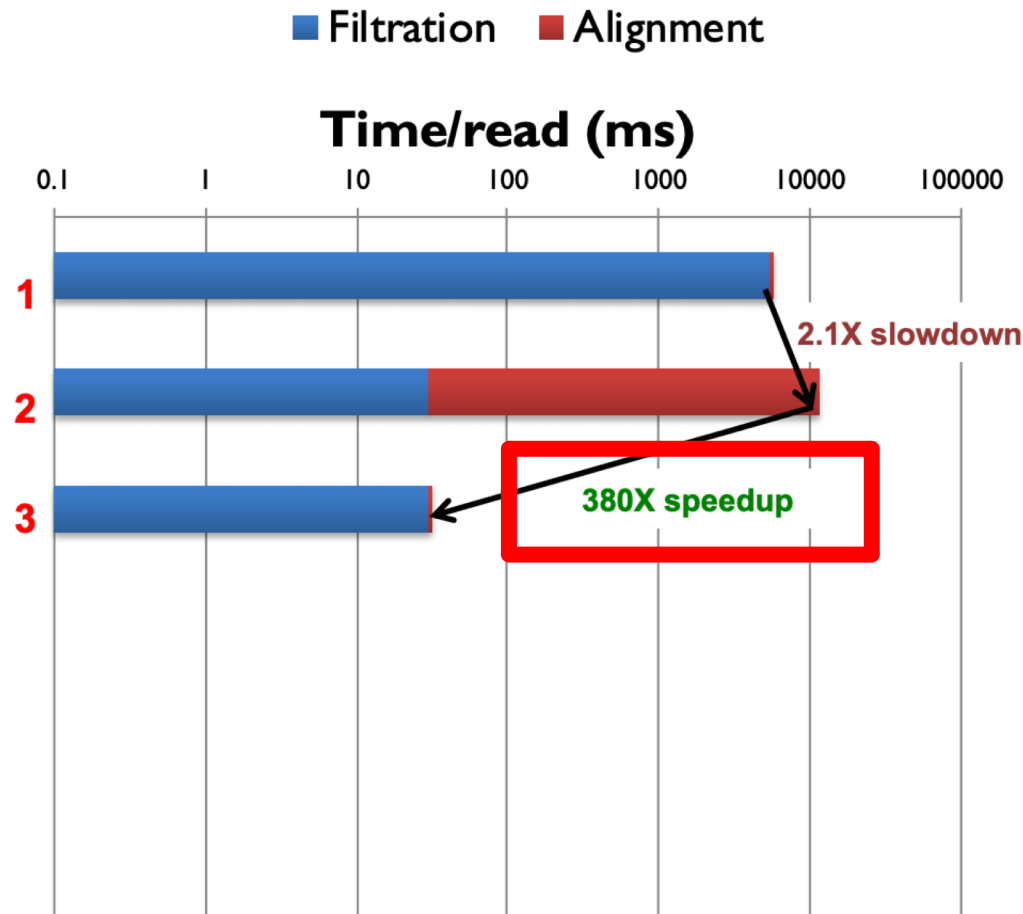
- **Solution:** Divide the table into **overlapping tiles** and compute them all **independently** using **systolic arrays**.
- Store the **trace** of each cell in an SRAM for **traceback**.



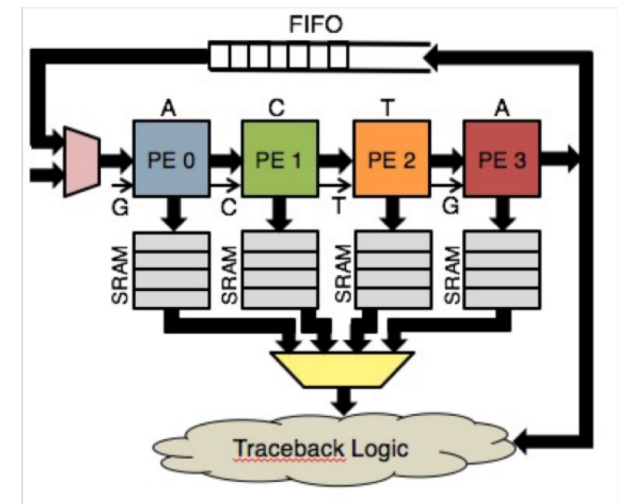
Implementation Details

- It is simulated for TSMC 40 nm CMOS process.
- 64 systolic arrays are working **concurrently**.
- 64 PEs (processing elements) in each systolic array.
- Each entry of the dynamic programming table accommodates 16-bit value.
- Each systolic array requires 128 KB SRAM (each PE =2 KB SRAM bank) for traceback purposes.

GACT Hardware vs. Software Speedup



1. Graphmap (software)
2. Replace by D-SOFT and GACT (software)
3. GACT hardware-acceleration



GACT Hardware vs. Edlib

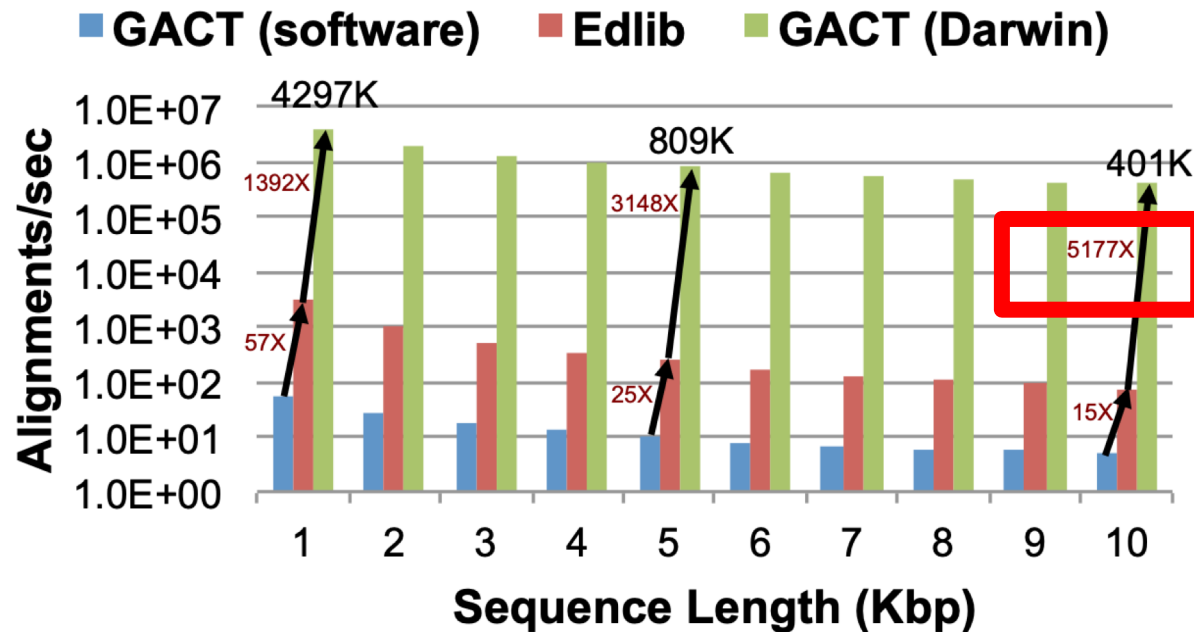


Figure 10: Throughput (alignments/second) comparison for different sequence lengths between a software implementation of GACT, Edlib library and the hardware-acceleration of GACT in Darwin.

More on Darwin

<https://github.com/gsneha26/Darwin-WGA>

Session 3A: Programmable Devices and Co-processors

ASPLOS'18, March 24–28, 2018, Williamsburg, VA, USA

Darwin: A Genomics Co-processor Provides up to 15,000× acceleration on long read assembly

Yatish Turakhia
Stanford University
yatisht@stanford.edu

Gill Bejerano
Stanford University
bejerano@stanford.edu

William J. Dally
Stanford University
NVIDIA Research
dally@stanford.edu

Yatish+ "Darwin: A genomics co-processor provides up to 15,000 x acceleration on long read assembly." *ASPLOS* 2018.

<http://bejerano.stanford.edu/papers/p199-turakhia.pdf>

Disclaimer on Darwin

- Darwin is NOT developed in **SAFARI group**, but we developed BitMAC that is now under review.
- BitMAC = new read alignment algorithm + PIM specialized accelerator.
- BitMAC provides 2.1x better throughput per unit area and 59.2x better throughput per unit power when compared with GACT of Darwin.

Conclusion on Ongoing Directions

- Read alignment can be **substantially accelerated** using **computationally inexpensive** and **accurate pre-alignment filtering** algorithms designed for specialized hardware.
- All the **three directions are used** by mappers today, but **filtering has replaced alignment as the bottleneck**.
- **Pre-alignment filtering** does *not* sacrifice any of the aligner capabilities, as it **does not modify or replace the alignment step**.

Agenda for Today

- Why Genome Analysis?
- What is Genome Analysis?
- How we Map Reads?
- What Makes Read Mapper Slow?
- Algorithmic & Hardware Acceleration
 - Seed Filtering Technique
 - Pre-alignment Filtering Technique
 - Read Alignment Acceleration
- Where is Read Mapping Going Next?

Where is Read Mapping Going Next?

Will 100% accurate genome-long reads alleviate/eliminate the need for read mapping?

Think about metagenomics, pan-genomics, ...

Where is Read Mapping Going Next?

nature genetics

Letter | [Open Access](#) | Published: 19 November 2018

Assembly of a pan-genome from deep sequencing of 910 humans of African descent

Rachel M. Sherman , Juliet Forman, [...] Steven L. Salzberg 

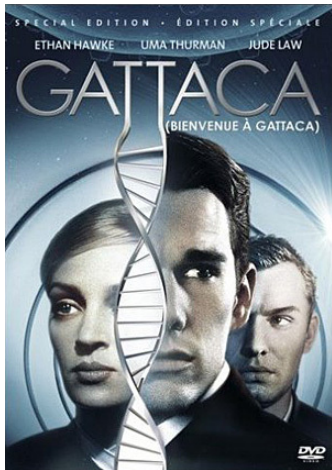
Nature Genetics **51**, 30–35(2019) | [Cite this article](#)

39k Accesses | **29** Citations | **875** Altmetric | [Metrics](#)

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

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).



1997

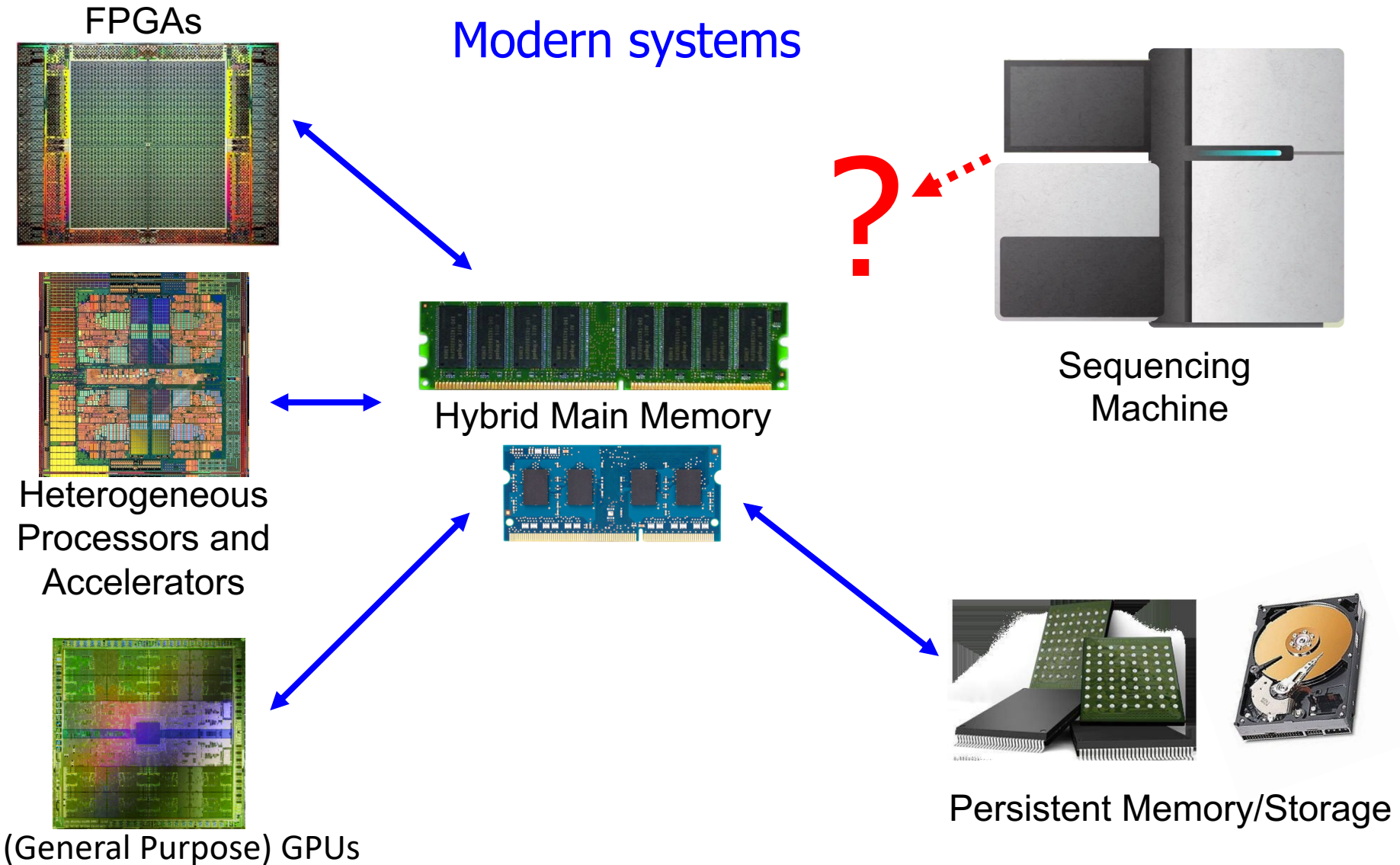


2015

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



Lecture Conclusion

- System design for bioinformatics is a critical problem
 - It has large scientific, medical, societal, personal implications
- This lecture is about accelerating a key step in bioinformatics: genome sequence analysis
 - In particular, read mapping
- Many bottlenecks exist in accessing and manipulating huge amounts of genomic data during analysis
- We cover various recent ideas to accelerate read mapping
 - A journey since September 2006

Acknowledgments

- Prof. Onur Mutlu, ETH Zurich
- Prof. Can Alkan, Bilkent University

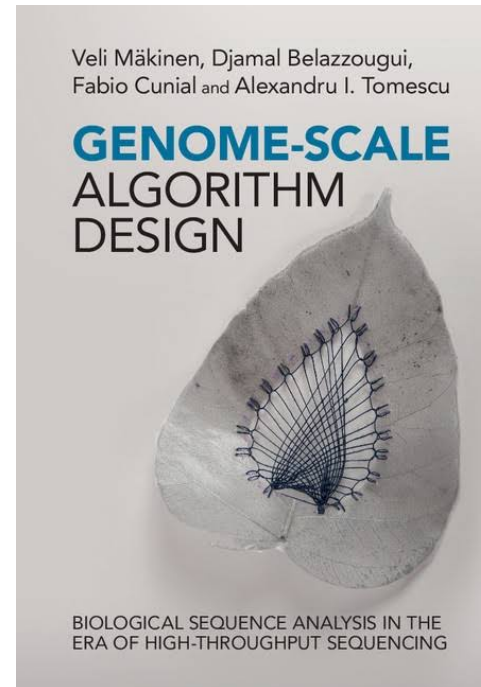
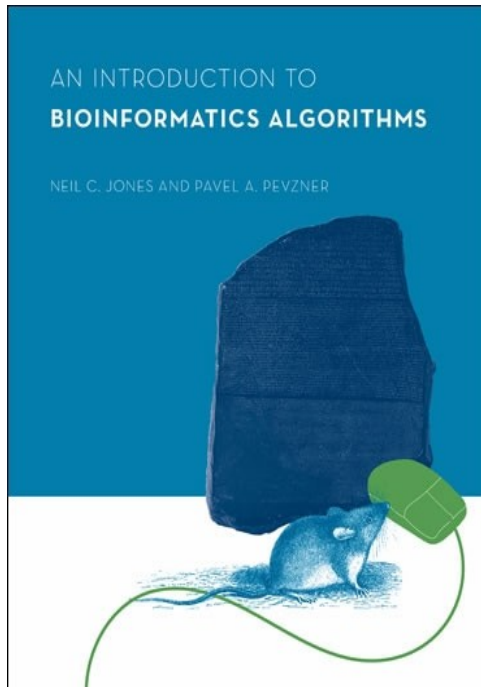
- Many colleagues and collaborators
 - Damla Senol Cali, Jeremie Kim, Hasan Hassan, Donghyuk Lee, Hongyi Xin, ...

- Funders:
 - NIH and Industrial Partners (Alibaba, AMD, Google, Facebook, HP Labs, Huawei, IBM, Intel, Microsoft, Nvidia, Oracle, Qualcomm, Rambus, Samsung, Seagate, VMware)

- All papers, source code, and more are at:
 - <https://people.inf.ethz.ch/omutlu/projects.htm>

Recommended Readings

- Jones, Neil C., Pavel A. Pevzner, and Pavel Pevzner. “[An introduction to bioinformatics algorithms](#),” MIT press, 2004.
- Mäkinen, Veli, Djamel Belazzougui, Fabio Cunial, and Alexandru I. Tomescu. “[Genome-scale algorithm design](#),” Cambridge University Press, 2015.



Accelerating Genome Analysis Using New Algorithms and Hardware Designs

Mohammed Alser

ETH Zurich

ALSERM@inf.ethz.ch

The University of Tokyo, Kashiwa Campus

18 December 2019

Firtina, Can, et al. "**Apollo: A Sequencing-Technology-Independent, Scalable, and Accurate Assembly Polishing Algorithm**." *arXiv preprint arXiv:1902.04341* (2019).

<https://arxiv.org/abs/1902.04341>

Apollo: A Sequencing-Technology-Independent, Scalable, and Accurate Assembly Polishing Algorithm

Can Firtina¹, Jeremie S. Kim^{1,2}, Mohammed Alser¹, Damla Senol Cali², A. Ercument Cicek³, Can Alkan^{3,*}, and Onur Mutlu^{1,2,3,*}

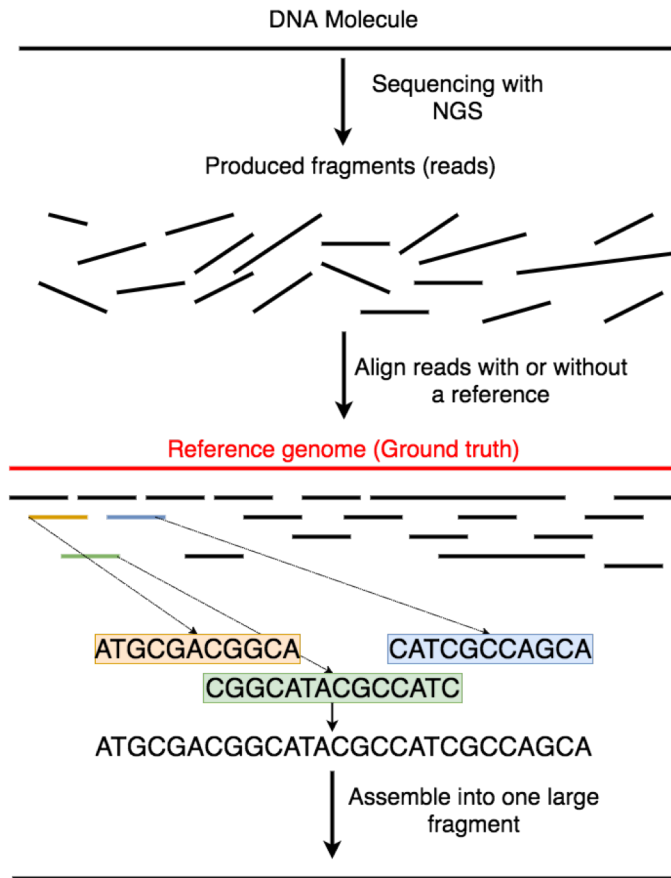
¹Department of Computer Science, ETH Zurich, Zurich 8092, Switzerland

²Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh 15213, PA, USA

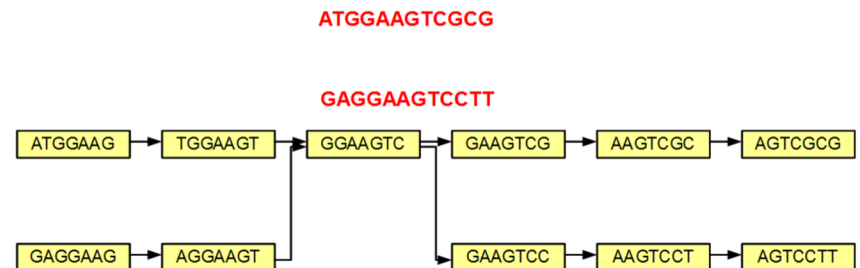
³Department of Computer Engineering, Bilkent University, Ankara 06800, Turkey

Constructing an assembly of reads

Alignment-based



Graph-based: De Bruijn Graphs



Apollo

■ Key observations:

- ❑ It **may not be possible** to construct the entire genome **using short reads** due to the complexity to find overlaps between short reads
- ❑ Re-assembling the **long reads** produce **erroneous genome**, which may cause incorrect findings in the further steps of the genome analysis
- ❑ Existing polishing tools **cannot polish large genomes**

■ Key idea:

- ❑ **Polishing the errors** in each contig of an **assembly individually** using all the available read sets (long + short reads) **within a single run.**

■ Key insights:

- ❑ **Errors are not random** and can be represented in a graph by assigning certain probabilities to resolve each error type at certain positions
- ❑ A profile hidden Markov model (pHMM) is a good fit to represent the actual contig as well as the possible errors that can take place after each basepair
- ❑ Aligning reads to a contig gives a clue about the differences between a contig and a read

■ Contribution

- ❑ First algorithm that can scale well **to polish large genomes** and that can use multiple read sets **from any sequencing technology within a single**

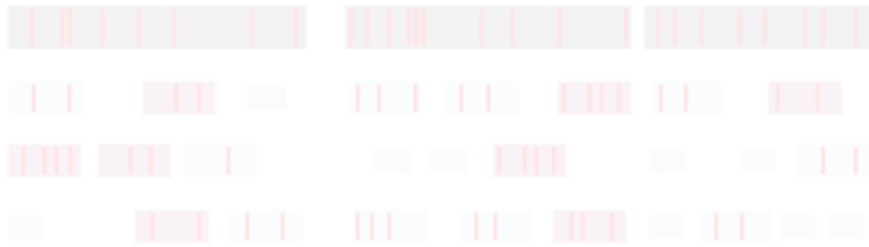
Assembly polishing pipeline

Input Preparation (External to Apollo)

Step 1: Assembly Construction

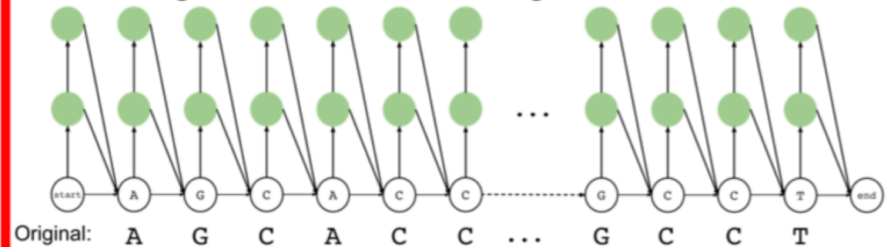


Step 2: Read-to-assembly alignment



Assembly Polishing (Internal to Apollo)

Step 3: Create a pHMM-graph per contig for correcting the errors in the contig



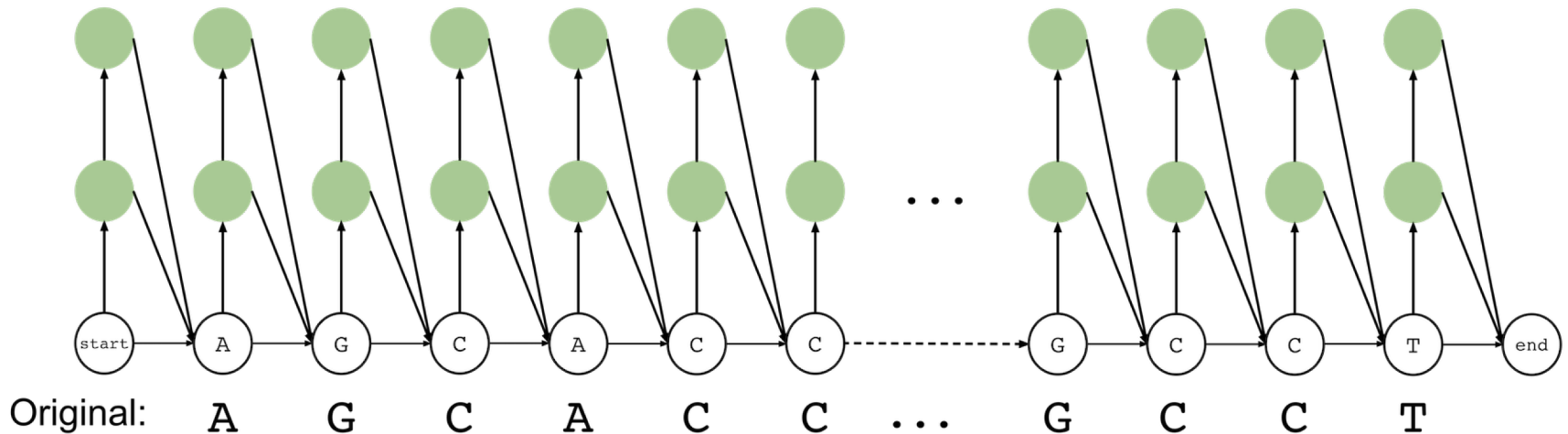
Step 4: The Forward-Backward algorithm updates the transition and emission probabilities of a pHMM-graph for each alignment to a contig

Step 5: Viterbi algorithm decodes the corrected contig



A Profile hidden Markov model

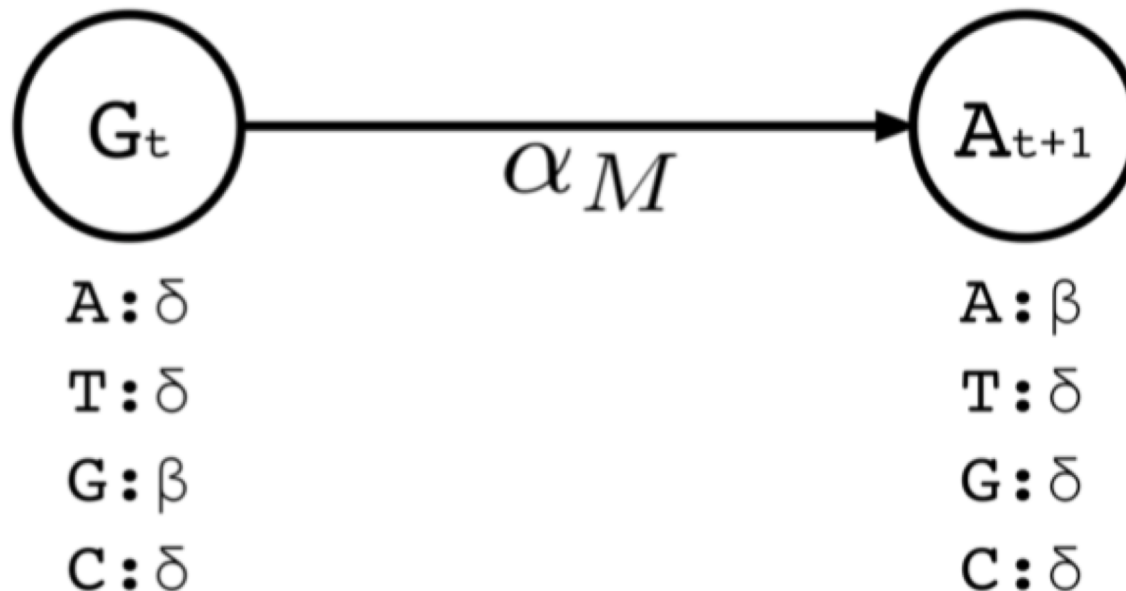
- Represent the contig “AGCACC...GCCT” in a pHMM-graph



- Each state emits (i.e., consumes or outputs) a single base when visited
- Correction:
 - Visiting insertion states to insert more bases between two bases in a contig
 - Skipping certain states to delete some bases
 - Emitting a different a different base than a base that is actually present at certain location (e.g., changing G to T at position 2)

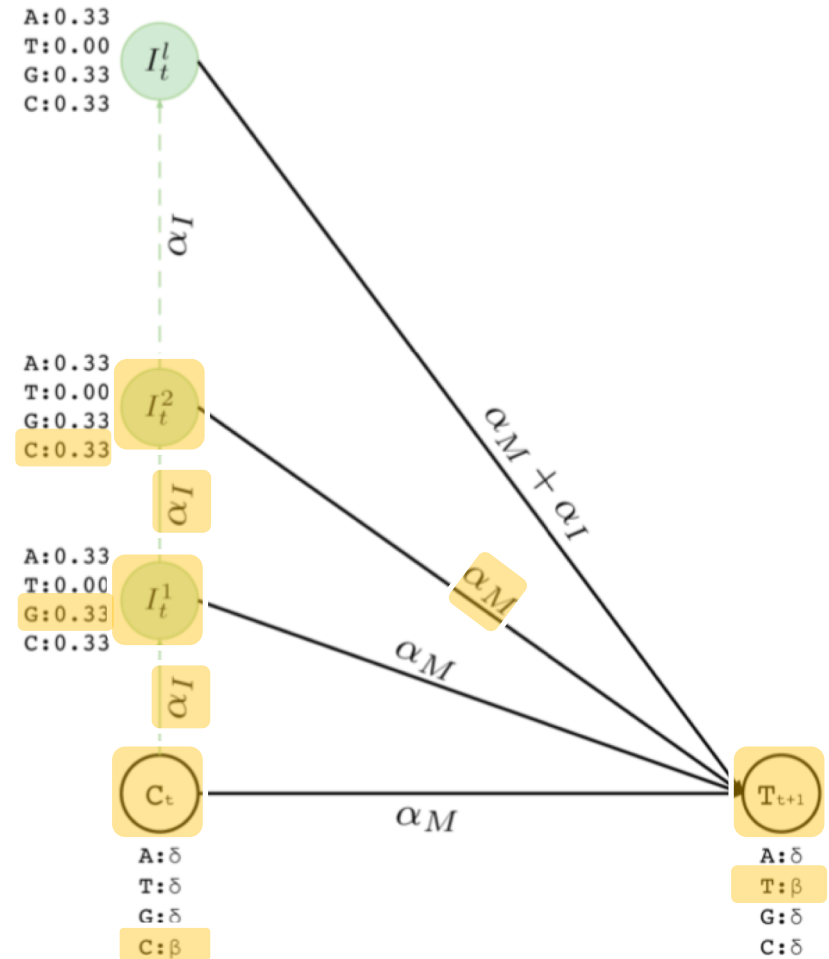
Resolving substitution errors or no error

- Match states for bases "GA" at positions t and $t+1$, respectively
- If no error: emit "G" at position t **with the probability of having no error**
- For substitution error, emit either A, T, or C at position t **with substitution error probability**
- All type of emission and transition probabilities are a parameter to Apollo



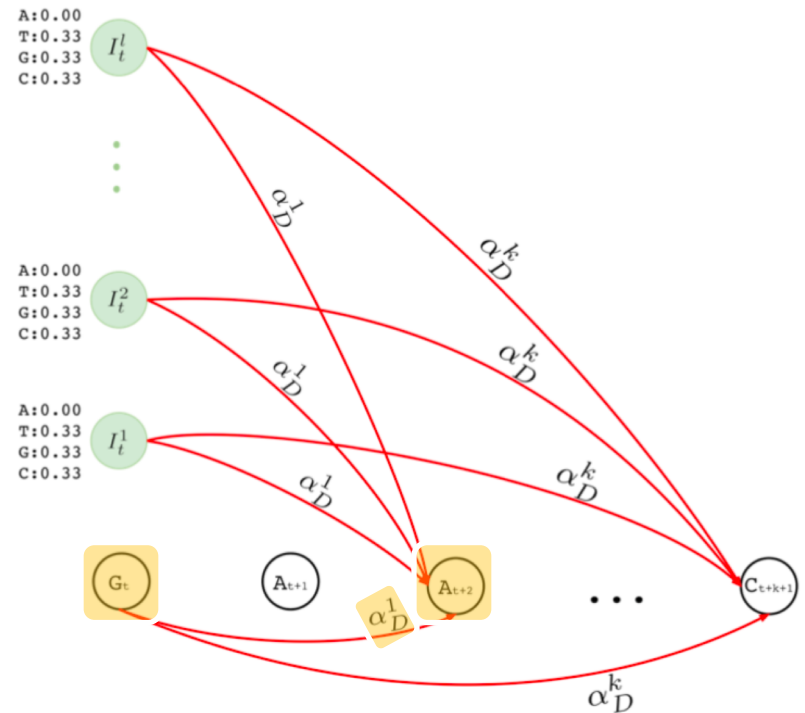
Resolving deletion errors

- Insertion states **to insert at most / many bases between two bases in a contig**
- To insert "GC" between "CT"
 - Visit match state at position t and *emit C*
 - Visit first *insertion state* after position t and *emit G with deletion error probability*
 - Visit second insertion state and *emit C with deletion error probability*
 - From second insertion state visit match state at position $t+1$ and *emit T*
 - Resulting sequence "CGCT"
- Maximum number of insertions is a parameter to Apollo



Resolving insertion errors

- Deletion transitions to delete one or many bases in a row
- To delete the first A in "GAA"
 - Visit match state at position t and *emit* G
 - Visit match state at position $t+2$ and emit A with *single insertion error probability*
 - Resulting sequence: "GA"
- Having single or more deletions in a row may not be necessarily equally likely
- Maximum number of deletions in a row is a parameter to Apollo



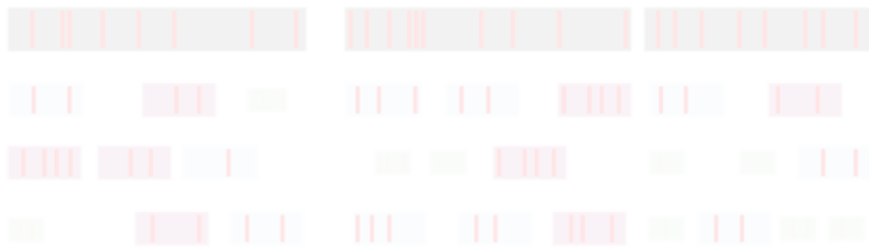
Assembly polishing pipeline

Input Preparation (External to Apollo)

Step 1: Assembly Construction



Step 2: Read-to-assembly alignment



Assembly Polishing (Internal to Apollo)

Step 3: Create a pHMM-graph per contig for correcting the errors in the contig



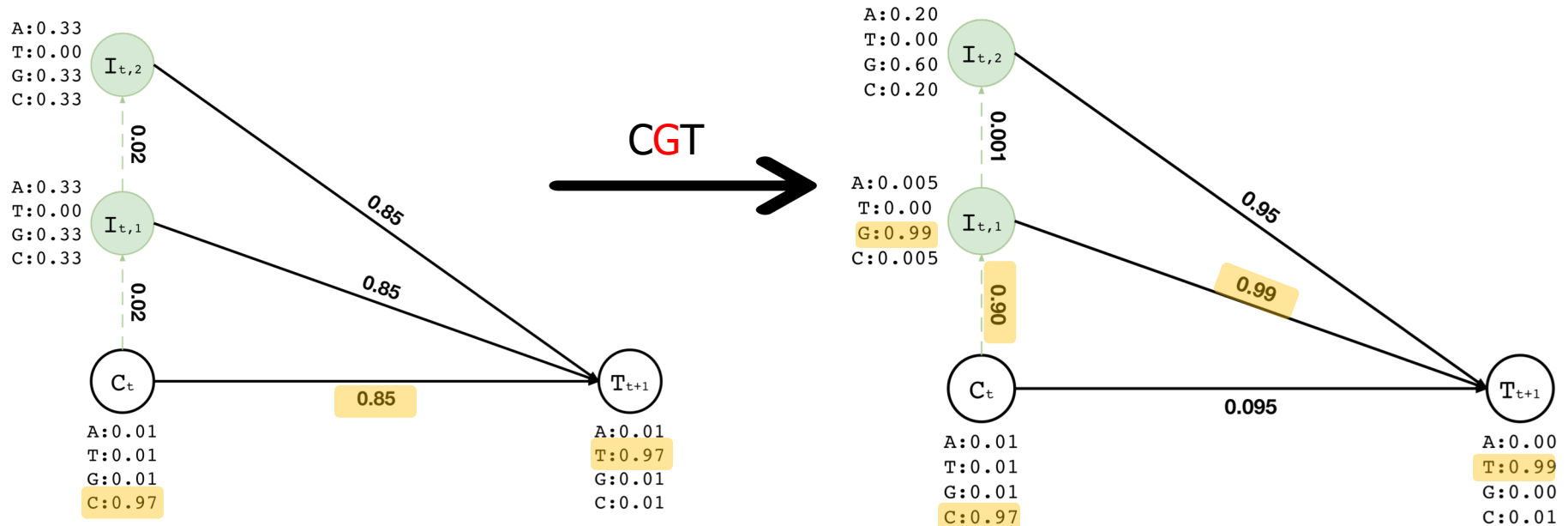
Step 4: The Forward-Backward algorithm updates the transition and emission probabilities of a pHMM-graph for each alignment to a contig

Step 5: Viterbi algorithm decodes the corrected contig



Training

- Training data:
 - Read aligned to the location t of a contig
- Assume we have the read "CGT" aligned to location t
- After training the corresponding region of the graph we would expect change in the probabilities **so that it will be likely to emit "CGT" somehow**



The Forward-Backward algorithm

- Calculating the likelihood of visiting a state to emit a certain character of a given sequence (i.e., aligned read)

- Forward calculation (F)

$$F_1(j) = \alpha_{0j} e_j(r[1]) \quad s.t. \quad j \in V_s, \quad E_{0j} \in E_s$$

$$F_t(j) = \sum_{i \in V_s} F_{t-1}(i) \alpha_{ij} e_j(r[t]) \quad j \in V_s, \quad 1 < t \leq m$$

- Backward calculation (B)

$$B_m(i) = \alpha_{i(m+1)} \quad i \in V_s, \quad E_{i(m+1)} \in E_s$$

$$B_t(i) = \sum_{j \in V_s} \alpha_{ij} e_j(r[t+1]) B_{t+1}(j) \quad j \in V_s, \quad 1 \leq t < m$$

- Backward calculation needs a starting point

Training: The Baum-Welch algorithm

- Expectation maximization step using the Baum-Welch algorithm

$$e_i^*(X) = \frac{\sum_{t=1}^m F_t(i) B_t(i) (r[t] == X)}{\sum_{t=1}^m F_t(i) B_t(i)} \quad \forall X \in \Sigma, \forall i \in V_s$$

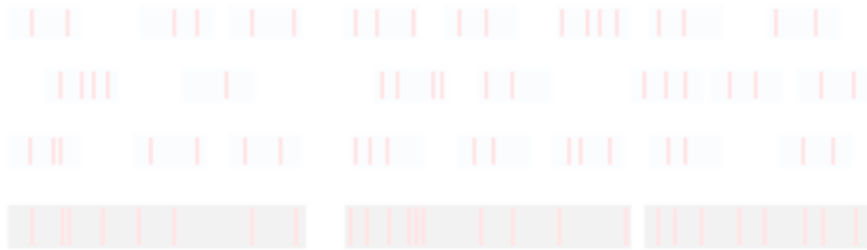
$$\alpha_{ij}^* = \frac{\sum_{t=1}^{m-1} \alpha_{ij} e_j(r[t+1]) F_t(i) B_{t+1}(j)}{\sum_{t=1}^{m-1} \sum_{x \in V_s} \alpha_{ix} e_x(r[t+1]) F_t(i) B_{t+1}(x)} \quad \forall E_{ij} \in E_s$$

- If there are multiple reads aligning to same region, we have multiple $F(i)$ for a position t
 - Take the average and use it as $F(i)$ for position t

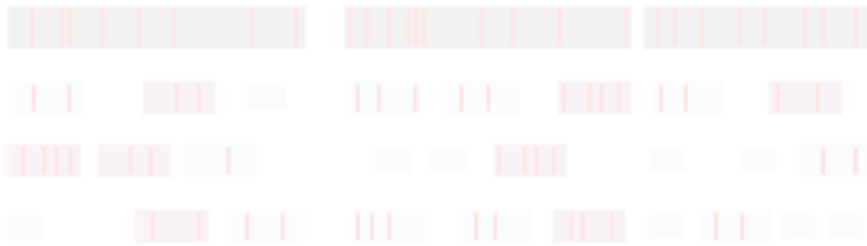
Assembly polishing pipeline

Input Preparation (External to Apollo)

Step 1: Assembly Construction



Step 2: Read-to-assembly alignment



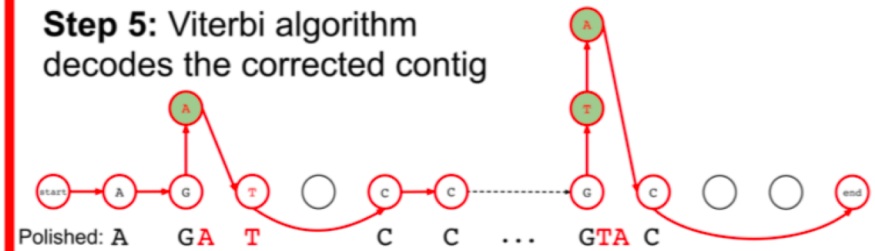
Assembly Polishing (Internal to Apollo)

Step 3: Create a pHMM-graph per contig for correcting the errors in the contig



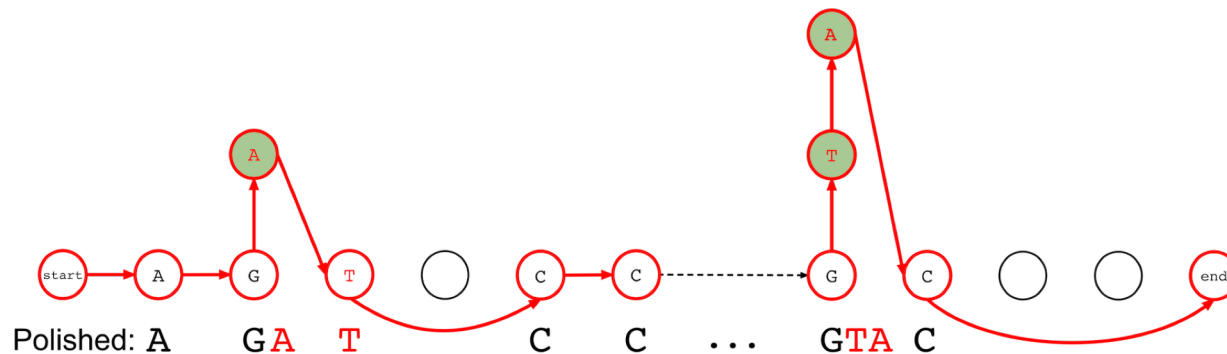
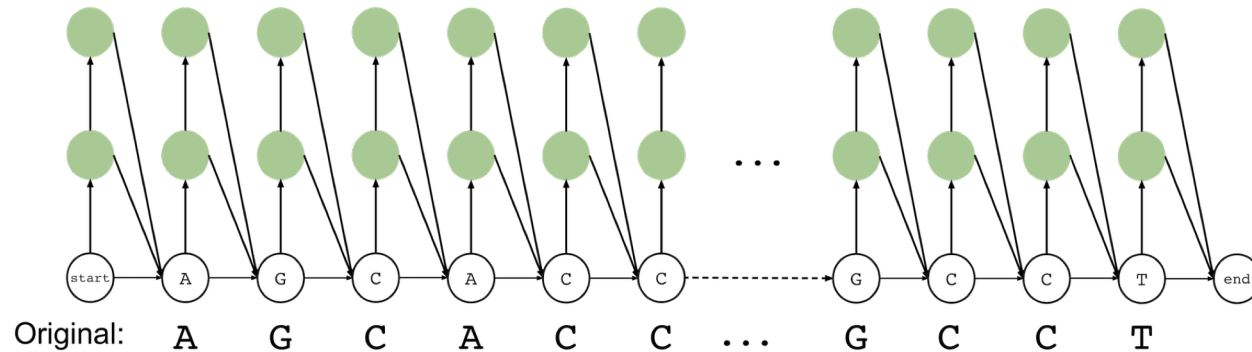
Step 4: The Forward-Backward algorithm updates the transition and emission probabilities of a pHMM-graph for each alignment to a contig

Step 5: Viterbi algorithm decodes the corrected contig



Inference: The Viterbi algorithm

- Our original contig before polishing was: "AGCACC...GCCT"
- After updating the probabilities, the most likely path from start to end reveals the corrected contig: "AGATCC...GTAC"



Inference: The Viterbi algorithm

- Decode the entire graph after training to have the corrected version of a contig

1. Initialization

$$v_1(j) = \hat{\alpha}_{start-j} \hat{e}_j(X') \quad \forall j \in V$$

$$b_1(j) = start \quad \forall j \in V$$

2. Recursion

$$v_t(j) = \max_{i \in V} v_{t-1}(i) \hat{\alpha}_{ij} \hat{e}_j(X') \quad \forall j \in V, 1 < t \leq T$$

$$b_t(j) = \operatorname{argmax}_{i \in V} v_{t-1}(i) \hat{\alpha}_{ij} \hat{e}_j(X') \quad \forall j \in V, 1 < t \leq T$$

3. Termination

$$v_T(end) = \max_{i \in V} v_T(i) \hat{\alpha}_{i-end}$$

$$b_T(end) = \operatorname{argmax}_{i \in V} v_T(i) \hat{\alpha}_{i-end}$$

Results

Data Sets

Data Set	Accession Number	Details
E.coli K-12 - ONT E.coli K-12 - Ground Truth	Loman Lab* GenBank NC_000913	164,472 reads (avg. 9,010bps, 319X coverage) via Metrichor Strain MG1655 (4,641Kbps)
E.coli O157 - PacBio E.coli O157 - Illumina E.coli O157 - Ground Truth	SRA SRR5413248 SRA SRR5413247 GenBank NJEX02000001	177,458 reads (avg. 4,724bps, 151X coverage) 11,856,506 paired-end reads (150bps each, 643X coverage) Strain FDAARGOS_292 (5,566Kbps)
E.coli O157:H7 - PacBio E.coli O157:H7 - Illumina E.coli O157:H7 - Ground Truth	SRA SRR1509640 SRA SRR1509643 GCA_000732965	76,279 reads (avg. 8,270bps, 112X coverage) 2,978,835 paired-end reads (250bps each, 265X coverage) Strain EDL933 (5,639Kbps)
Yeast S288C - PacBio Yeast S288C - Illumina Yeast S288C - Ground Truth	SRA ERR165511(8-9), ERR1655125 SRA ERR1938683 GCA_000146055.2	296,485 reads (avg. 5,735bps, 140X coverage) 3,318,467 paired-end reads (150bps each, 82X coverage) Strain S288C (12,157Kbps)
Human CHM1 - PacBio Human CHM1 - Ground Truth	SRA SRR130433(1-5) GCA_000306695.2	912,421 reads (avg. 8,646bps, 2.6X coverage) 3.04Gbps
Human HG002 - PacBio Human HG002 - Illumina Human HG002 - Ground Truth	SRA SRR2036(394-471), SRR203665(4-9) SRA SRR17664(42-59) GCA_001542345.1	15,892,517 reads (avg. 6,550bps, 35X coverage) 222,925,733 paired-end reads (148bps each, 22X coverage) Ashkenazim trio - Son (2.99Gbps)

Applicability of the Polishing Algorithms to Large Genomes

Aligner	Sequencing Tech. of the Reads	Polishing Algorithm	Runtime	Memory (GB)
Minimap2	PacBio (35X)	Apollo	227h 12m 15s	62.91
BWA-MEM	PacBio (35X)	Apollo	198h 41m 15s	58.60
Minimap2	PacBio (35X)	Racon	N/A	N/A
BWA-MEM	PacBio (35X)	Racon	N/A	N/A
pbalign	PacBio (35X)	Quiver	N/A	N/A
Minimap2	PacBio (8.9X)	Apollo	55h 38m 44s	44.99
BWA-MEM	PacBio (8.9X)	Apollo	41h 38m 27s	45.00
Minimap2	PacBio (8.9X)	Racon	2h 48m 25s	54.13
BWA-MEM	PacBio (8.9X)	Racon	1h 36m 39s	51.55
pbalign	PacBio (8.9X)	Quiver	N/A	N/A
Minimap2	Illumina (22X)	Apollo	96h 22m 16s	101.12
BWA-MEM	Illumina (22X)	Apollo	102h 01m 57s	107.06
Minimap2	Illumina (22X)	Racon	N/A	N/A
BWA-MEM	Illumina (22X)	Racon	N/A	N/A
Minimap2	Illumina (22X)	Pilon	N/A	N/A
BWA-MEM	Illumina (22X)	Pilon	N/A	N/A

Benefits of using a hybrid set of reads

Data Set	First Run	Second Run	Aligned Bases (%)	Accuracy	Polishing Score	Runtime	Memory (GB)
E.Coli O157			99.94	0.9998	0.9992	43m 53s	3.79
E.Coli O157	Apollo (Hybrid)		99.94	0.9999	0.9993	8h 16m 08s	13.85
E.Coli O157	Racon (PacBio)	Racon (Illumina)	99.94	0.9994	0.9988	21m 44s	22.65
E.Coli O157	Racon (PacBio)	Racon (PacBio)	99.94	0.9984	0.9978	4m 58s	2.43
E.Coli O157	Racon (PacBio)	Pilon (Illumina)	99.40	0.9989	0.9829	12m 14s	8.51
E.Coli O157	Pilon (Illumina)	Pilon (Illumina)	99.94	0.9999	0.9993	4m 10s	11.40
E.Coli O157	Pilon (Illumina)	Racon (PacBio)	99.94	0.9986	0.9980	4m 58s	11.40
E.Coli O157	Quiver (PacBio)	Pilon (Illumina)	99.94	0.9998	0.9992	5m 01s	7.50
E.Coli O157	Quiver (PacBio)	Racon (PacBio)	99.94	0.9986	0.9980	5m 13s	2.48
E.Coli O157:H7			100.00	0.9998	0.9998	43m 19s	3.39
E.Coli O157:H7	Apollo (Hybrid)		100.00	0.9999	0.9999	5h 58m 05s	8.86
E.Coli O157:H7	Racon (PacBio)	Racon (Illumina)	100.00	0.9995	0.9995	9m 43s	6.56
E.Coli O157:H7	Racon (PacBio)	Racon (PacBio)	100.00	0.9970	0.9970	5m 36s	2.24
E.Coli O157:H7	Racon (PacBio)	Pilon (Illumina)	100.00	0.9996	0.9996	10m 23s	6.41
E.Coli O157:H7	Pilon (Illumina)	Pilon (Illumina)	100.00	0.9998	0.9998	35m 12s	10.79
E.Coli O157:H7	Pilon (Illumina)	Racon (PacBio)	100.00	0.9996	0.9996	6m 04s	10.75
Yeast S288C			99.89	0.9998	0.9987	1h 20m 39s	6.24
Yeast S288C	Apollo (Hybrid)		99.89	0.9998	0.9987	11h 08m 41s	6.38
Yeast S288C	Racon (PacBio)	Racon (Illumina)	99.89	0.9994	0.9983	38m 21s	6.93
Yeast S288C	Racon (PacBio)	Racon (PacBio)	99.89	0.9949	0.9938	49m 52s	6.93
Yeast S288C	Racon (PacBio)	Pilon (Illumina)	99.89	0.9992	0.9981	26m 25s	14.25
Yeast S288C	Pilon (Illumina)	Pilon (Illumina)	99.89	0.9998	0.9987	1m 10s	11.85
Yeast S288C	Pilon (Illumina)	Racon (PacBio)	99.89	0.9960	0.9949	21m 42s	11.85

Using a set of reads from a single sequencing technology

- Still comparable performance for smaller genomes even when a single set of reads used

Sequencing Tech. of the Assembly	Assembler	Aligner	Sequencing Tech. of the Reads	Polishing Algorithm	Aligned Bases (%)	Accuracy	Polishing Score	Runtime	Memory (GB)
PacBio	Miniasm	-	-	-	94.93	0.9000	0.8544	1m 48s	10.03
PacBio	Miniasm	Minimap2	PacBio	Apollo	98.49	0.9798	0.9650	2h 27m 49s	7.07
PacBio	Miniasm	Minimap2	PacBio	Pilon	96.43	0.9528	0.9188	1h 31m 32s	17.68
PacBio	Miniasm	Minimap2	PacBio	Racon	99.35	0.9951	0.9886	2m 13s	2.44
PacBio	Miniasm	pbalign	PacBio	Quiver	99.80	0.9993	0.9973	7m 31s	0.51
PacBio	Miniasm	Minimap2	Illumina	Apollo	97.61	0.9816	0.9581	4h 25m 17s	9.22
PacBio	Miniasm	Minimap2	Illumina	Pilon	96.52	0.9775	0.9435	32m 48s	18.60
PacBio	Miniasm	Minimap2	Illumina	Racon	96.45	0.9876	0.9525	14m 09s	21.57
PacBio	Miniasm	BWA-MEM	Illumina	Apollo	96.62	0.9738	0.9409	3h 32m 45s	9.21
PacBio	Miniasm	BWA-MEM	Illumina	Pilon	96.13	0.9693	0.9318	31m 21s	18.45
PacBio	Miniasm	BWA-MEM	Illumina	Racon	96.90	0.9813	0.9509	12m 05s	20.85
PacBio	Canu	-	-	-	99.94	0.9998	0.9992	43m 53s	3.79
PacBio	Canu	Minimap2	PacBio	Apollo	99.94	0.9997	0.9991	3h 42m 03s	8.82
PacBio	Canu	Minimap2	PacBio	Racon	99.94	0.9986	0.9980	2m 17s	2.34
PacBio	Canu	pbalign	PacBio	Quiver	99.94	0.9998	0.9992	7m 06s	0.20
PacBio	Canu	BWA-MEM	Illumina	Apollo	99.94	0.9999	0.9993	4h 49m 15s	11.05
PacBio	Canu	BWA-MEM	Illumina	Pilon	99.94	0.9998	0.9992	2m 05s	11.40
PacBio	Canu	BWA-MEM	Illumina	Racon	99.94	0.9999	0.9993	14m 58s	21.04
PacBio (30X)	Miniasm*	-	-	-	-	-	-	-	-
PacBio (30X)	Canu	-	-	-	99.98	0.9981	0.9979	21m 03s	3.70
PacBio (30X)	Canu	Minimap2	PacBio (30X)	Apollo	99.98	0.9982	0.9980	43m 32s	8.00
PacBio (30X)	Canu	Minimap2	PacBio (30X)	Racon	99.98	0.9980	0.9978	15s	0.59
PacBio (30X)	Canu	Minimap2	PacBio (30X, Corr.)	Apollo	99.97	0.9976	0.9973	46m 10s	7.99
PacBio (30X)	Canu	Minimap2	PacBio (30X, Corr.)	Racon	99.98	0.9983	0.9981	7s	0.37
PacBio (30X)	Canu	BWA-MEM	Illumina	Apollo	99.98	0.9997	0.9995	4h 48m 31s	10.35
PacBio (30X)	Canu	BWA-MEM	Illumina	Pilon	99.98	0.9998	0.9996	3m 03s	8.52
PacBio (30X)	Canu	BWA-MEM	Illumina	Racon	99.98	0.9997	0.9995	14m 42s	21.04

Takeaways

- For large genomes, Apollo is the only algorithm that can scale well to use available data set
- Polish Canu-generated assemblies with a hybrid set of reads if the intention is to produce most reliable assembly
- The pHMM-graph proposed in Apollo is very flexible
 - Change the parameters according to the error profile of a sequencing machine
 - Decide whether to chunk a pHMM-graph or not during decoding
- Not good in terms of the run time
- Viterbi and Forward-Backward calculations per state very simple but yet serial in the current implementation

Apollo: A Sequencing-Technology-Independent, Scalable, and Accurate Assembly Polishing Algorithm

Can Firtina¹, Jeremie S. Kim^{1,2}, Mohammed Alser¹, Damla Senol Cali², A. Ercument Cicek³, Can Alkan^{3,*}, and Onur Mutlu^{1,2,3,*}

¹Department of Computer Science, ETH Zurich, Zurich 8092, Switzerland

²Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh 15213, PA, USA

³Department of Computer Engineering, Bilkent University, Ankara 06800, Turkey

SAFARI-Intel Mini Workshop

Can Firtina

May 8th, 2019

SAFARI

ETH zürich