CSHL Biological Data Science 2024

Rawsamble

Overlapping and Assembling Raw Nanopore Signals Using a Hash-based Seeding Mechanism

Can Firtina

Maximilian Mordig Harun Mustafa Sayan Goswami

Nika Mansouri Ghiasi Stefano Mercogliano Furkan Eris

Joël Lindegger Andre Kahles Onur Mutlu

SAFARI ETH zürich BIOMEDICAL INFORMATICS

MAX PLANCK INSTITUTE FOR INTELLIGENT SYSTEMS USZ Universitäts Spital Zürich SIB Swiss Institute of Bioinformatics

SAFARI

Executive Summary

Problem: Existing solutions **cannot** interpret raw signals directly **for reference-free applications**

Goal: Enable raw signal analysis **without a reference genome**

Key Contributions:

1. **Rawsamble: the first mechanism** that can find **overlapping pairs** between raw nanopore signals
2. **First *de novo* assemblies ever constructed** directly from raw signal overlaps **without basecalling**
3. **A new assembler** to build and output the assemblies of signals

Key Results: Across 5 genomes of varying sizes, Rawsamble provides

- **Average speedup of 16×** compared to Dorado (Fast model) + minimap2
- **37%** of overlapping pairs **shared with the minimap2 overlaps**
- **Unitigs up to 400× longer than the average read length**

Outline

Background

Rawsamble Mechanism

Evaluation

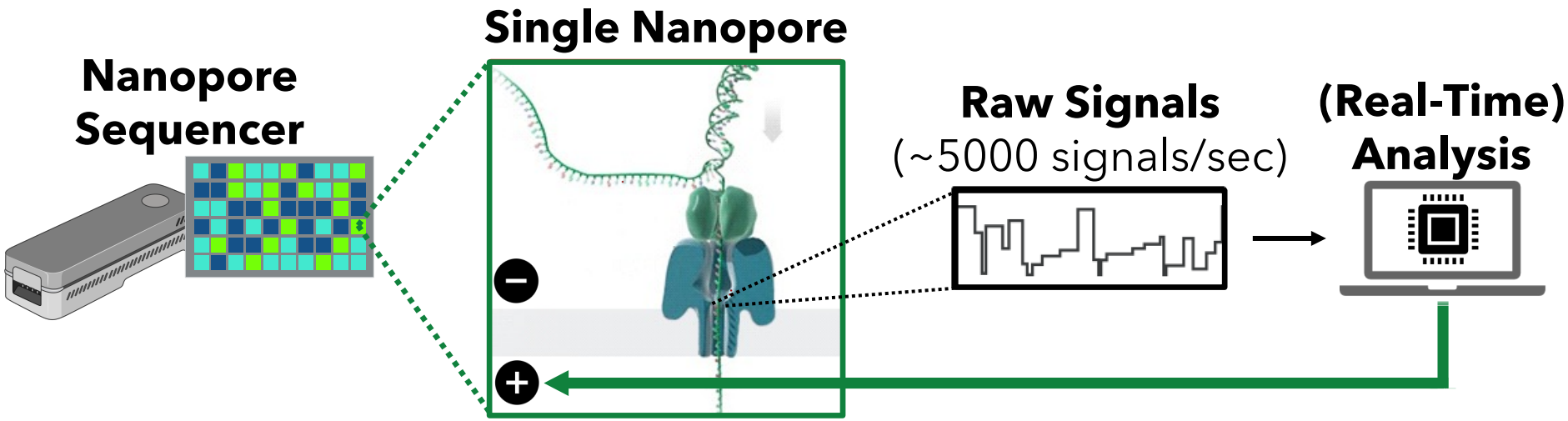
Conclusion

Nanopore Sequencing

Nanopore Sequencing: a widely used sequencing technology

- Can sequence large fragments of nucleic acid molecules
- Offers high throughput
- Cost-effective
- Enables **real-time and portable genome analysis**

Nanopore Sequencing – How it Works



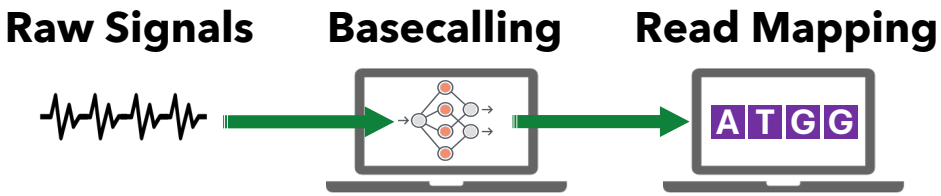
Raw Signals: Ionic current measurements generated at a certain **throughput**

(Real-Time) Analysis: Analyzing raw signals **instantly as they are generated**

Real-Time Decisions: Stopping sequencing **early** based on real-time analysis

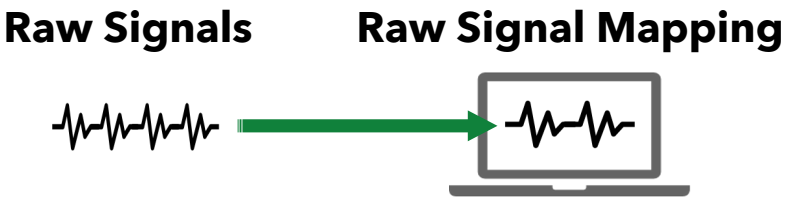
Analyzing Raw Nanopore Signals

Traditional: Translating (**basecalling**) signals to bases **before** analysis



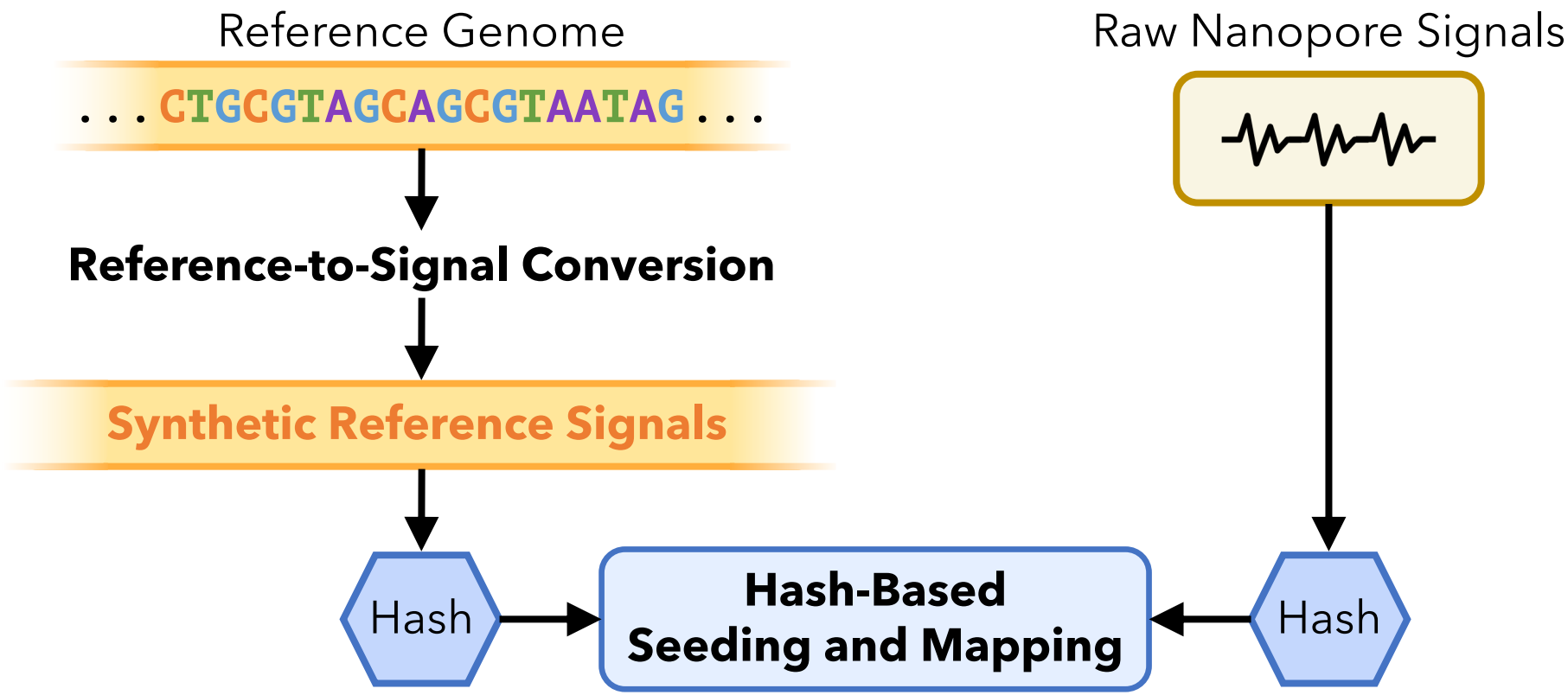
- ✓ Basecalled sequences are **less noisy** than raw signals
- ✓ **Many analysis tools** use basecalled sequences
- ✗ **Costly and power-hungry** computational requirements

Recent Works: Directly analyzing signals **without basecalling**



- ✓ **Efficient analysis** with better scalability and portability
- ✓ Raw signals retain **more information** than just bases
- ✗ **Lack of established tools** for downstream analysis

The State-of-the-Art Raw Signal Mapper



Synthetic signals are mainly **free from noise**

A reference genome must exist for mapping

RawHash [Firtina+, ISMB/ECCB'23]; RawHash2 [Firtina+, Bioinformatics'24]

The State-of-the-Art Raw Signal Mapper

Reference Genome

...CTGCGTAGCAGCGTAATAG...

Raw Nanopore Signals

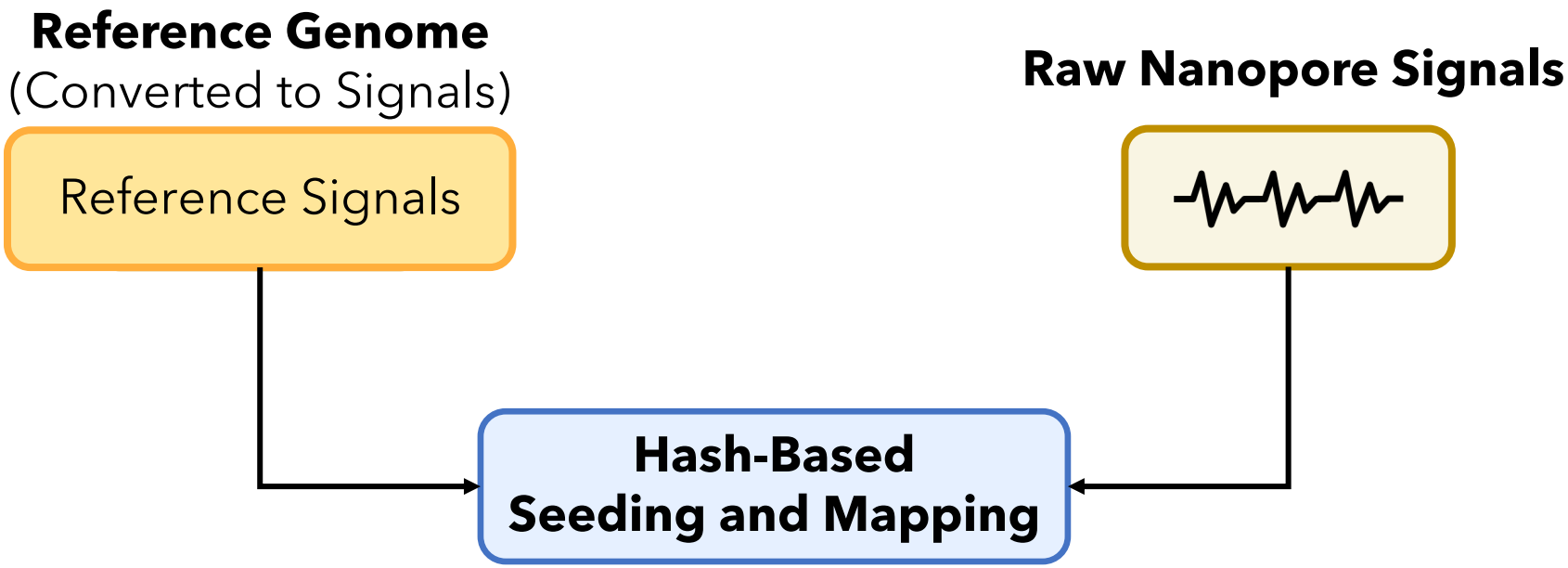
Existing solutions **cannot** analyze raw signals directly **without a reference genome**

Synthetic signals are mainly **free from noise**

A reference genome must exist for mapping

RawHash [Firtina+, ISMB/ECCB'23]; RawHash2 [Firtina+, Bioinformatics'24]

Beyond Reference Mapping: Overlapping

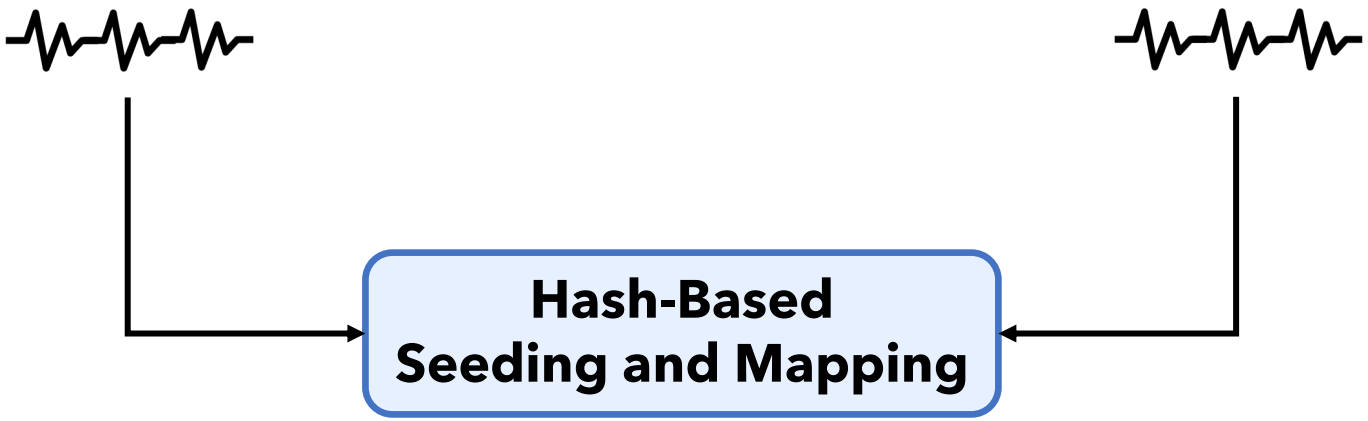


Challenge: Reference genomes are not always available

Assembly: Constructing genome from **overlapping reads**

Existing solutions cannot find overlapping reads **without basecalling**

Challenges with Overlapping Raw Signals



Challenge: Identifying hash matches **when both signals are noisy**

Challenge: Finding **many** useful overlapping pairs (all-vs-all overlapping)

Challenge: Generating **long paths** from useful overlaps

Outline

Background

Rawsamble Mechanism

Evaluation

Conclusion

Goal

Enable raw signal analysis
without a reference genome

Rawsamble

The first mechanism that can **perform all-vs-all overlapping from raw signals**

First *de novo* assemblies ever constructed directly from raw signal overlaps **without basecalling**

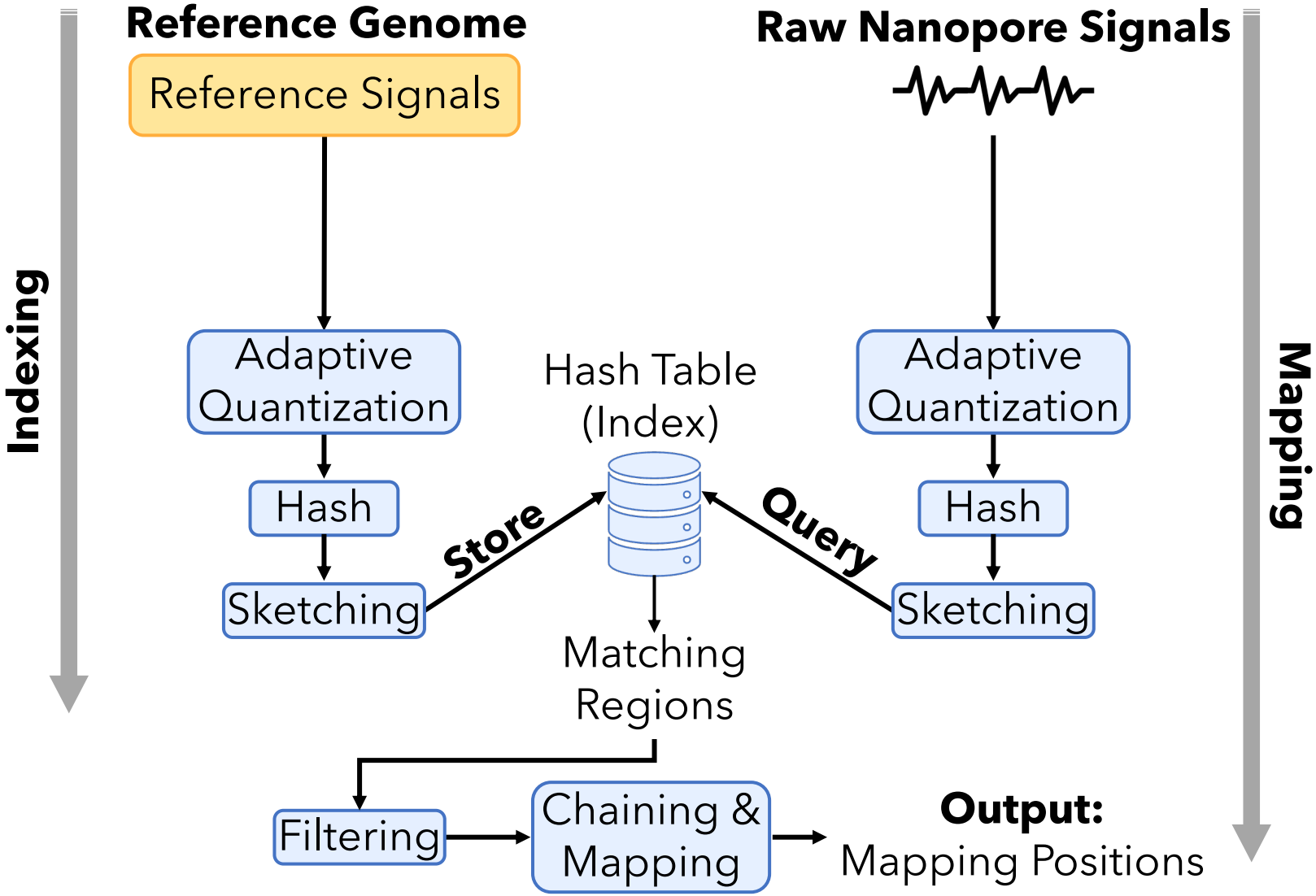
A new assembler to build and output the assemblies from raw signals

Rawsamble Key Ideas

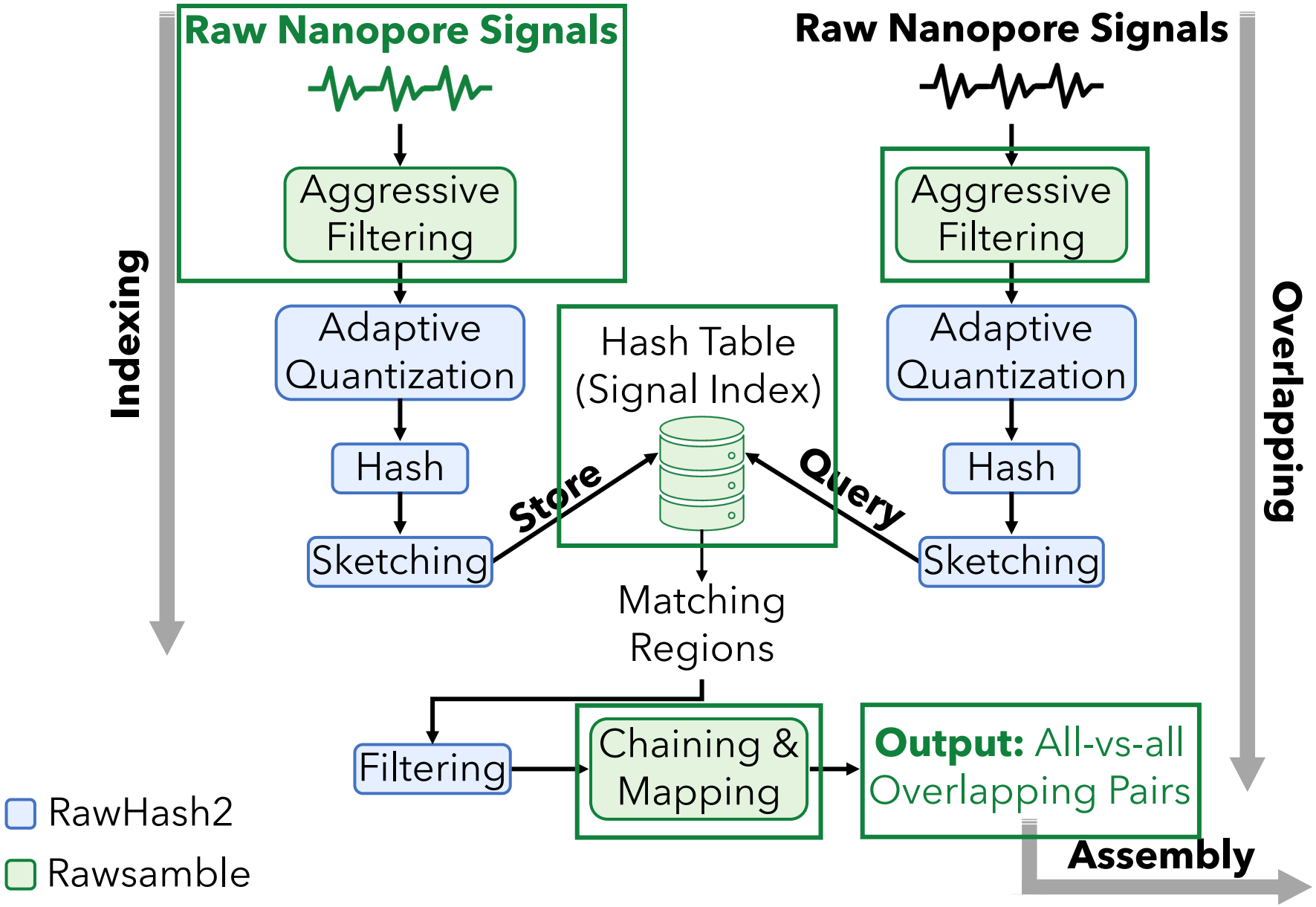
Build on the existing state-of-the-art raw signal mapper: **RawHash2**

Extend RawHash2 to **support overlapping**

Raw Signal Mapping with RawHash2



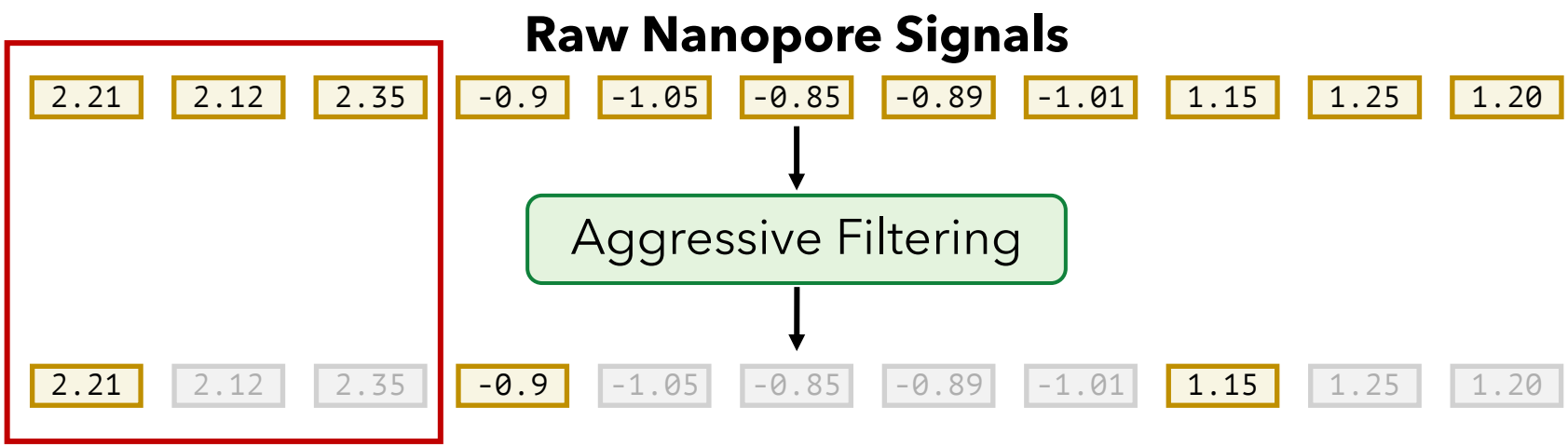
Integrating Rawsamble into RawHash2



Indexing using Raw Signals

1. Constructing the hash table index **from raw nanopore signals**
   - Reference-to-signal conversion **is mainly free from noise** (e.g., stay errors)
   - Indexed raw nanopore signals **are not free from noise**

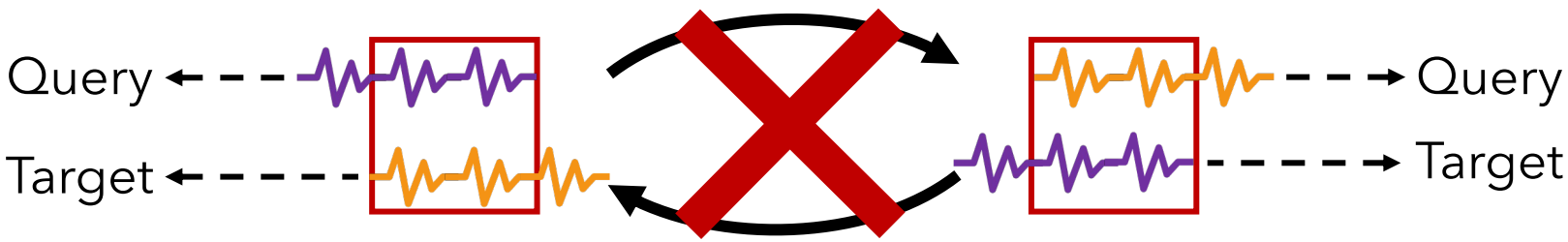
2. **Aggressively filtering** consecutive and similar signals to **substantially reduce noise** at the cost of data loss



Chaining and Outputting Overlaps

3. **Adjusting the minimum chaining score** to avoid false chains
   - All-vs-all overlapping tends to find a larger number of seed hits than mapping to a reference genome
   - Minimum score for a chain during overlapping is set to be **~5× larger than mapping**
   - **All such chains are reported** (instead of a single best mapping)

4. **Avoid cyclic overlaps** with deterministic comparisons



Outline

Background

Rawsamble Mechanism

Evaluation

Conclusion

Evaluation Methodology

- Rawsamble is integrated into **RawHash2** [Firtina+, Bioinformatics'24]

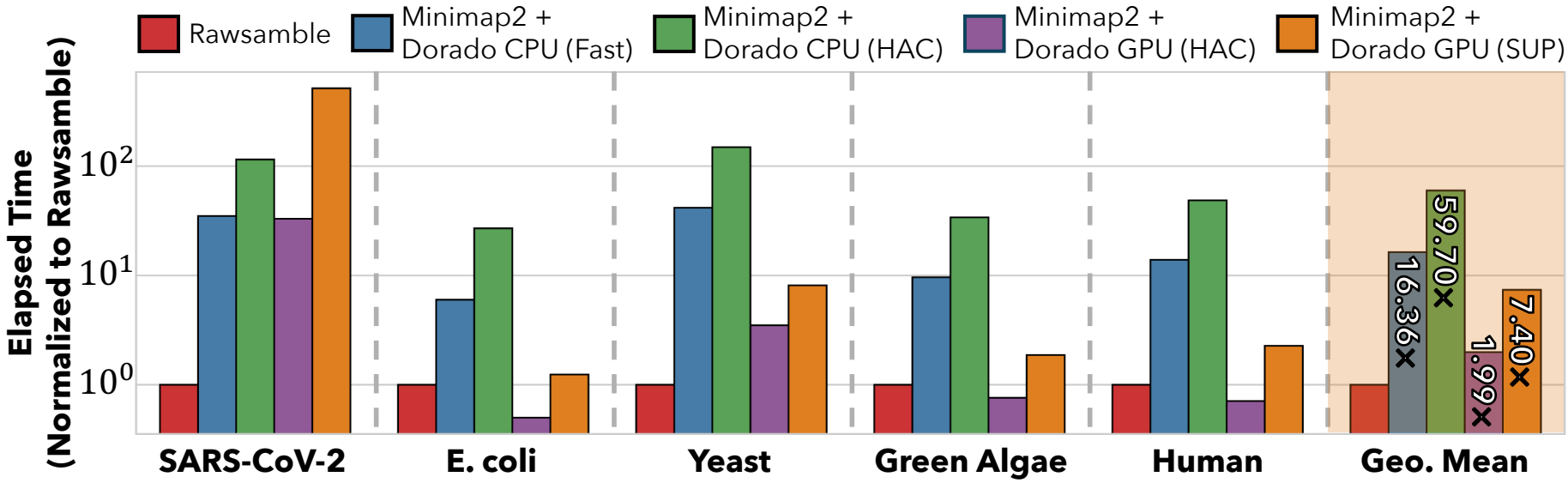
- Compared to the **minimap2** [Li, Bioinformatics'18] overlaps (forward strand)
  - Basecalling with **Dorado**'s various models (using CPUs & GPUs)

- **Use case** for raw signal overlapping:
  - De novo assembly construction using **miniasm** [Li, Bioinformatics'16]
  - New exciting directions to be discussed as future work

- **Evaluation metrics:**
  - Overall runtime when performing all-vs-all overlapping
  - Percentage of shared overlaps between tools
  - Assembly statistics

- **5 real datasets** with
  - Various **coverage** (0.6× – 445×) and
  - **Genome lengths** (viral to human genomes)

Normalized Runtime Results

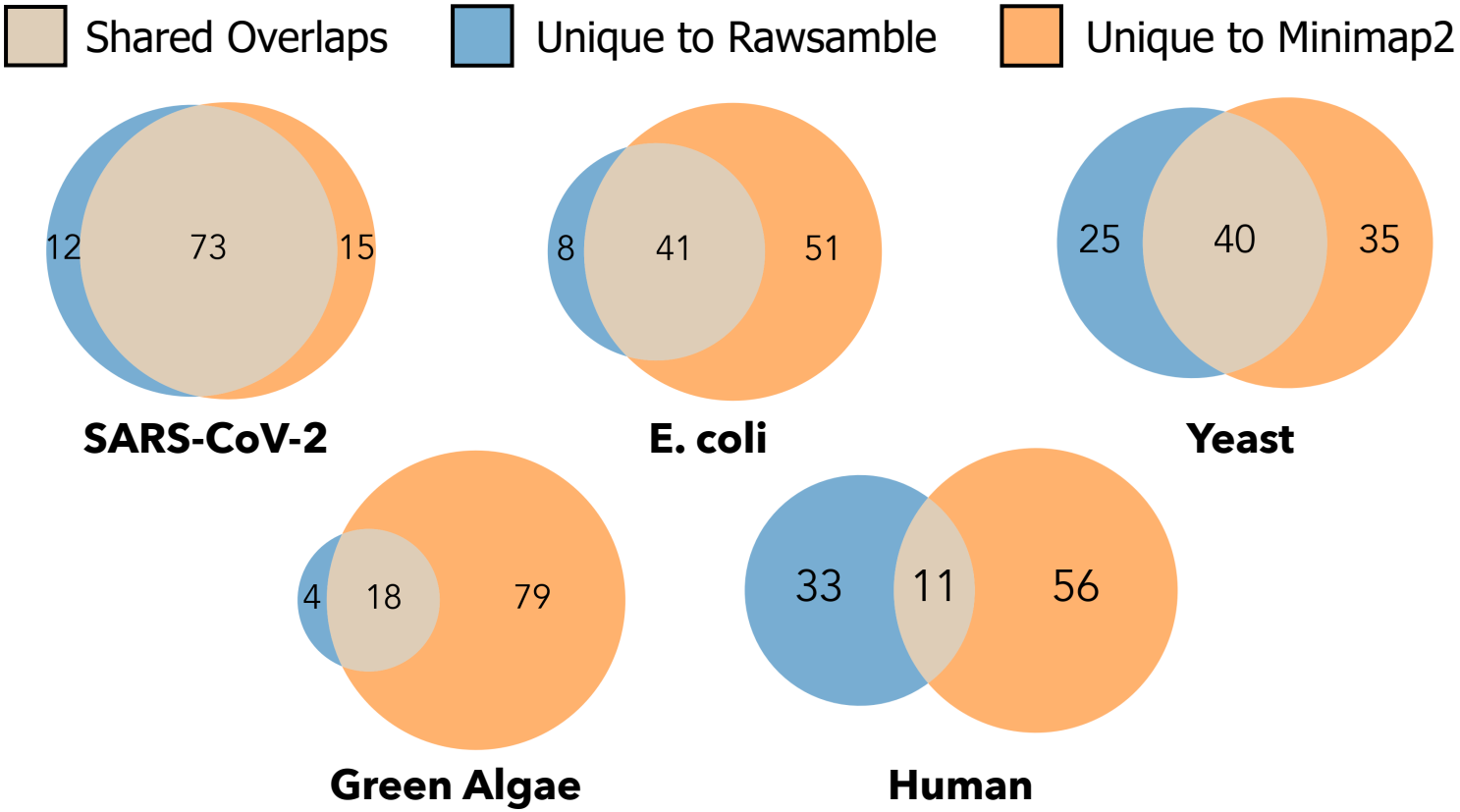


Compared to the fastest CPU model (Fast):
Average speedup of **16.36**×

Compared to the conventional GPU model (HAC):
Average speedup of **1.99**×

All-vs-All Overlapping Statistics

- **Percentage of overlapping pairs that are:**

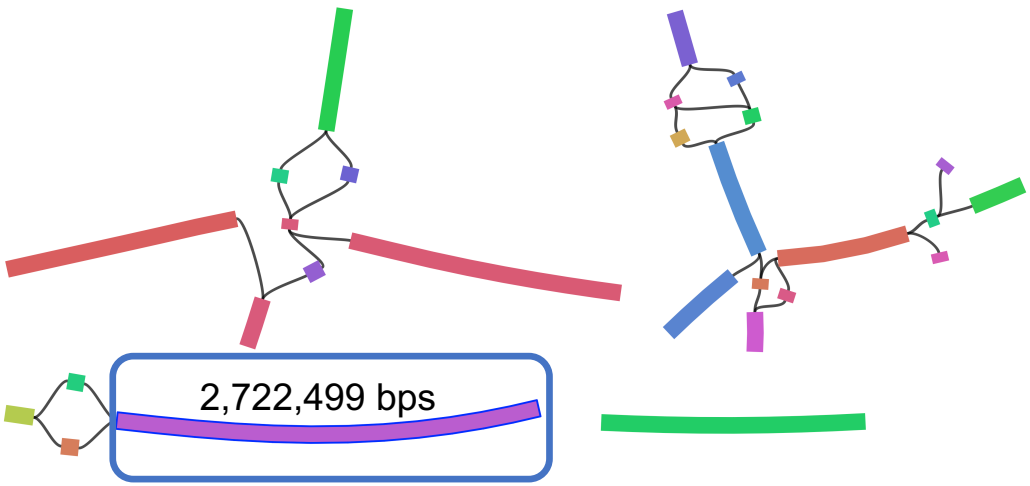


~37% of overlapping pairs are shared with the minimap2 overlaps

How to evaluate the usefulness of these overlaps?

de novo Assemblies from Raw Signals

E. coli Assembly
(From the Rawsamble Overlaps)



First *de novo* assemblies ever constructed from raw signal overlaps **without basecalling**

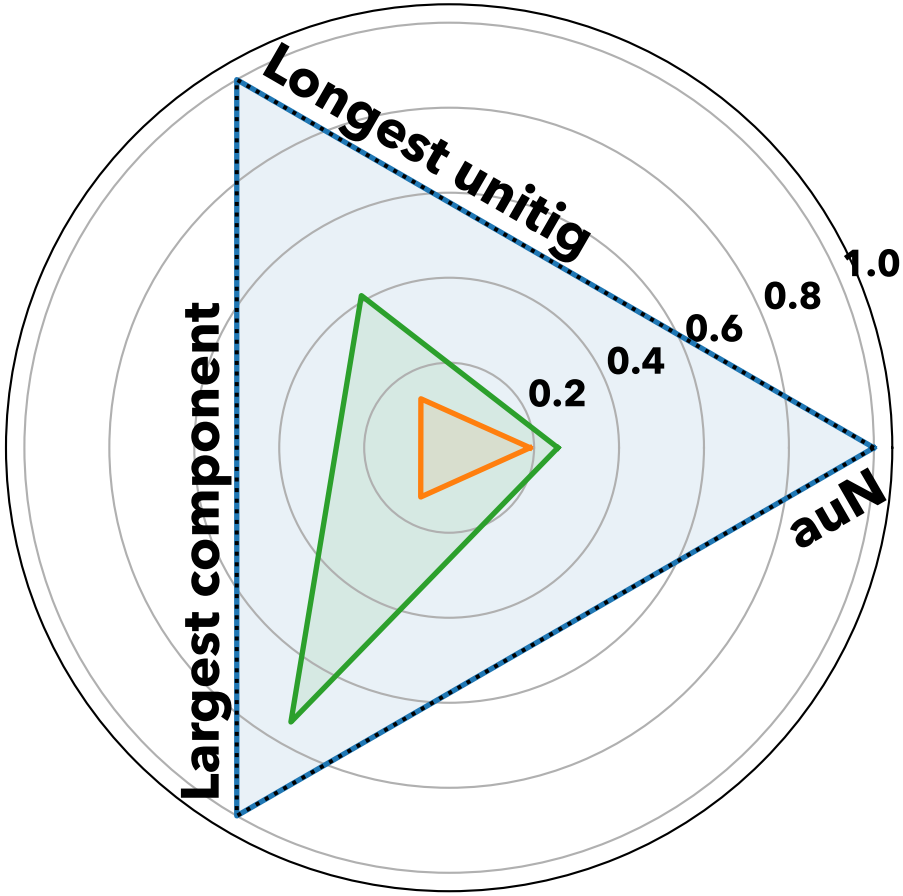
Contigs of **half the E. coli genome length** (~2.7 Mbases): **~400× longer than the average read length**

Future work: Utilize the overlap and assembly information **when training and using basecallers**

Low-Coverage Human Genome Assembly

- Results are shown **relative to the best result from each metric**
  - **Metrics:** auN, longest unitig, largest connected unitigs (component)
  - **Coverage: 0.6x**

Rawsamble Minimap2 Flye



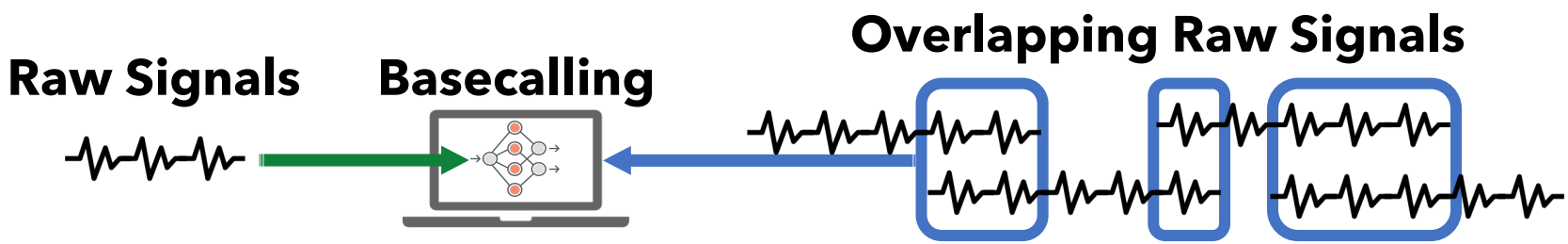
Rawsamble leads to **better contiguity** than using basecalled reads **at a low coverage dataset**

Can **the richer information in raw signals improve assembly quality** where basecalled analysis falls short?

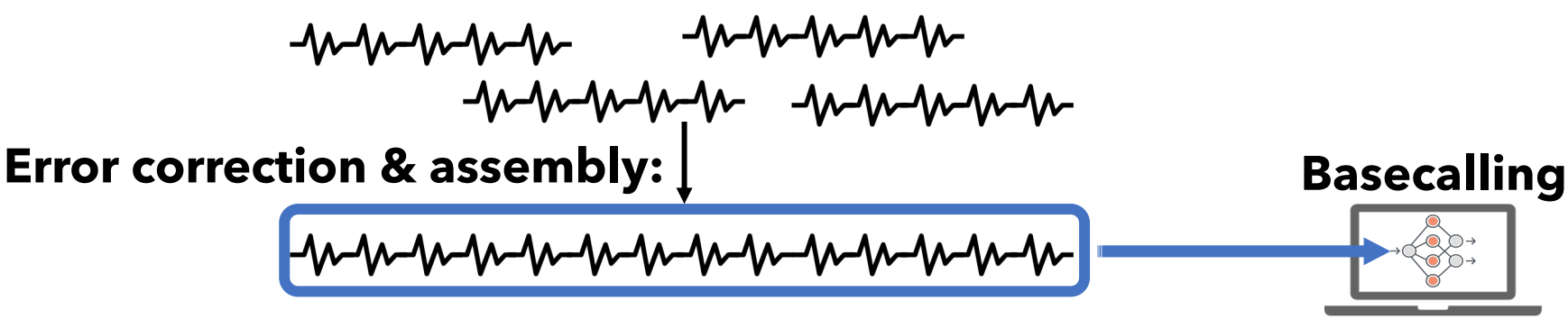
New Directions in Raw Signal Analysis

✓ Constructing (and analyzing) *de novo* assemblies

Utilizing the overlap information for more accurate (and faster) basecalling



Utilizing the constructed assembly for basecalling



Rawsamble

- Can Firtina, Maximilian Mordig, Harun Mustafa, Sayan Goswami, Nika Mansouri Ghiasi, Stefano Mercogliano, Furkan Eris, Joël Lindegger, Andre Kahles, and Onur Mutlu
  "Rawsamble: Overlapping and Assembling Raw Nanopore Signals using a Hash-based Seeding Mechanism"
  arXiv, Oct 2024
  [Source Code]

arXiv

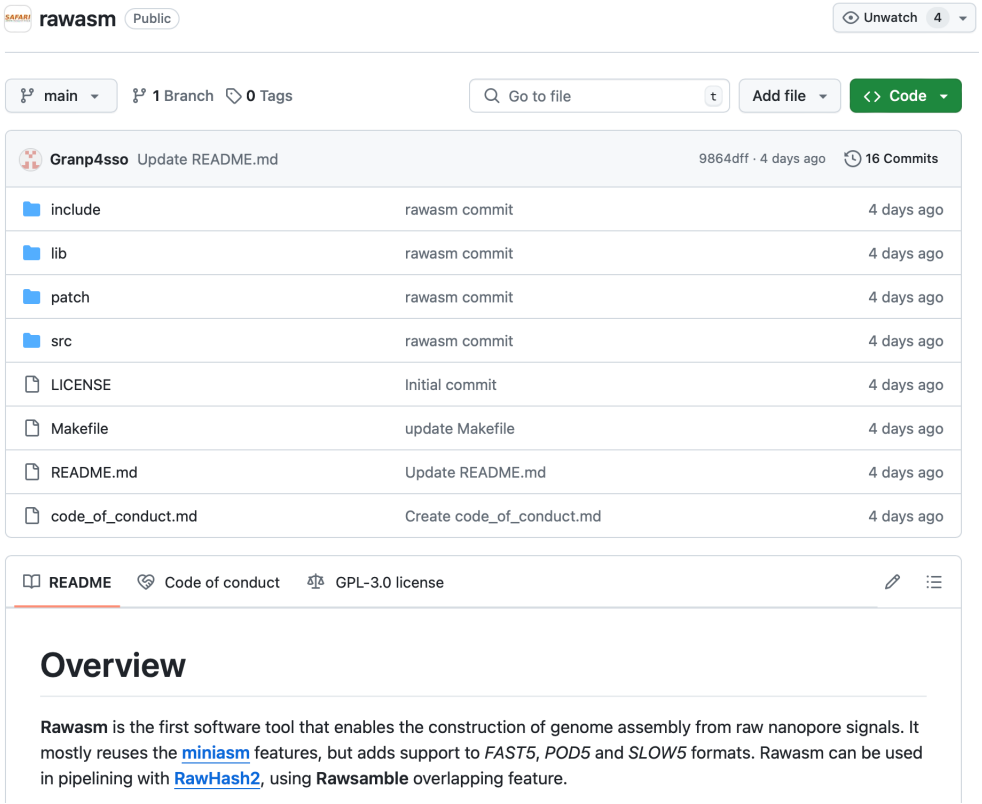
Source code

Rawsamble: Overlapping and Assembling Raw Nanopore Signals using a Hash-based Seeding Mechanism

Can Firtina[1] Maximilian Mordig[1,2] Harun Mustafa[1,3,4] Sayan Goswami[1] Nika Mansouri Ghiasi[1]
Stefano Mercogliano[1] Furkan Eris[1] Joël Lindegger[1] Andre Kahles[1,3,4] Onur Mutlu[1]
[1]*ETH Zurich* [2]*Max Planck Institute for Intelligent Systems*
[3]*University Hospital Zurich* [4]*Swiss Institute of Bioinformatics*

Rawasm: Raw Signal Assembler [Beta]

- **Slightly modified version of miniasm**
  - To output assembled raw signals instead of basecalled sequences
- Supports **all major raw signal file formats**
  - FAST5, POD5, S/BLOW5 file formats
- Still in a testing phase: Feedback is appreciated!



https://github.com/CMU-SAFARI/rawasm

Outline

Background

Rawsamble Mechanism

Evaluation

Conclusion

Conclusion

Key Contributions:

1. **Rawsamble: the first mechanism** that can find **overlapping pairs** between raw nanopore signals
2. **First *de novo* assemblies ever constructed** directly from raw signal overlaps **without basecalling**
3. **A new assembler** to build and output the assemblies of signals

Key Results: Across 5 genomes of varying sizes, Rawsamble provides

- **16× average speedup** compared to Dorado (Fast model) + minimap2
- **37%** of overlapping pairs **shared with the minimap2 overlaps**
- **Unitigs up to 400× longer than the average read length**

Many opportunities for analyzing raw nanopore signals:

- **Indexing is cheap:** Future use cases with the on-the-fly index construction
- We should rethink the algorithms to perform downstream analysis **fully using raw signals**
- **We should rethink the basecalling approaches by integrating raw signal analysis**

CSHL Biological Data Science 2024

Rawsamble

Overlapping and Assembling Raw Nanopore Signals Using a Hash-based Seeding Mechanism

Can Firtina

Maximilian Mordig Harun Mustafa Sayan Goswami

Nika Mansouri Ghiasi Stefano Mercogliano Furkan Eris

Joël Lindegger Andre Kahles Onur Mutlu

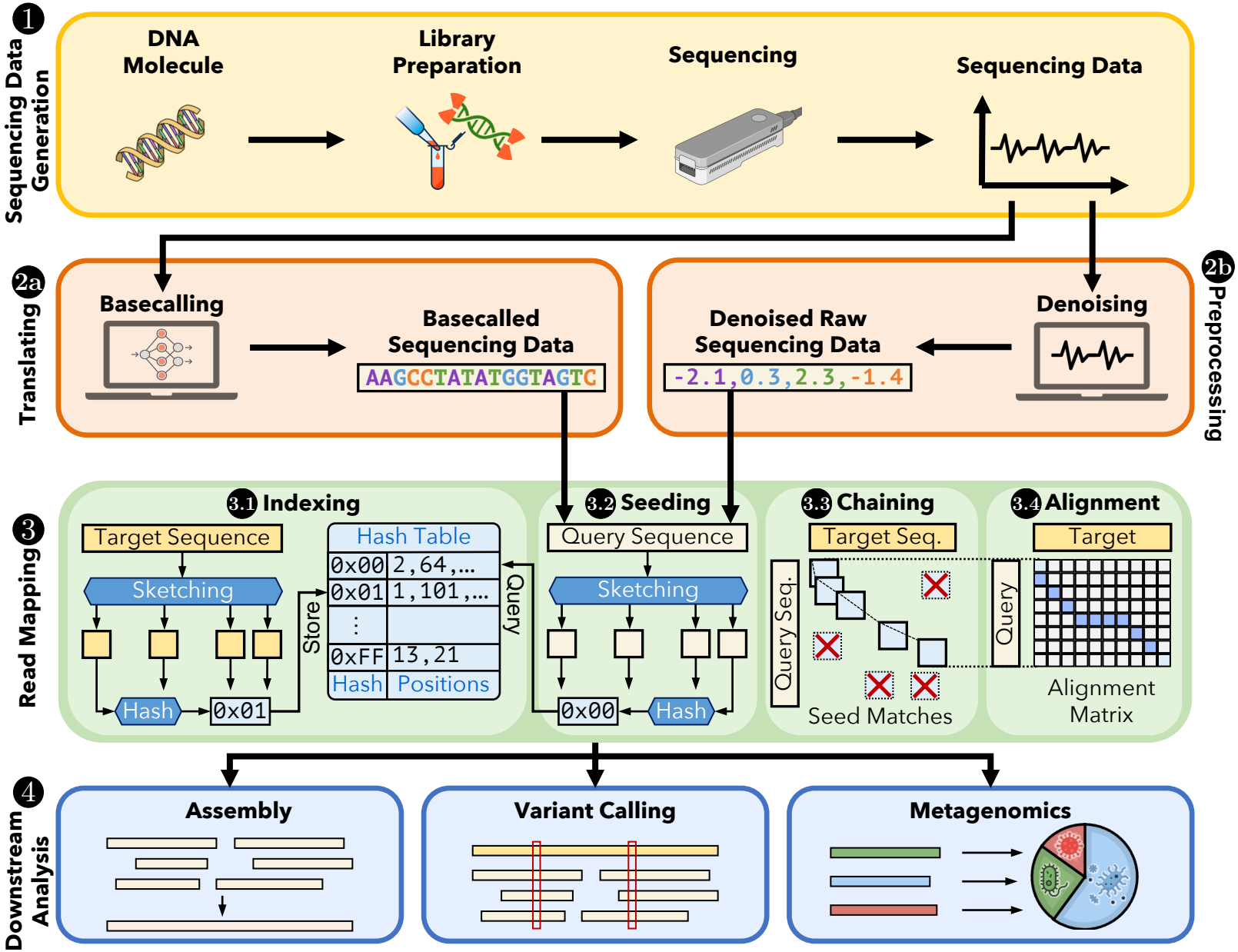
SAFARI ETH zürich BIOMEDICAL INFORMATICS

MAX PLANCK INSTITUTE FOR INTELLIGENT SYSTEMS USZ Universitäts Spital Zürich SIB Swiss Institute of Bioinformatics

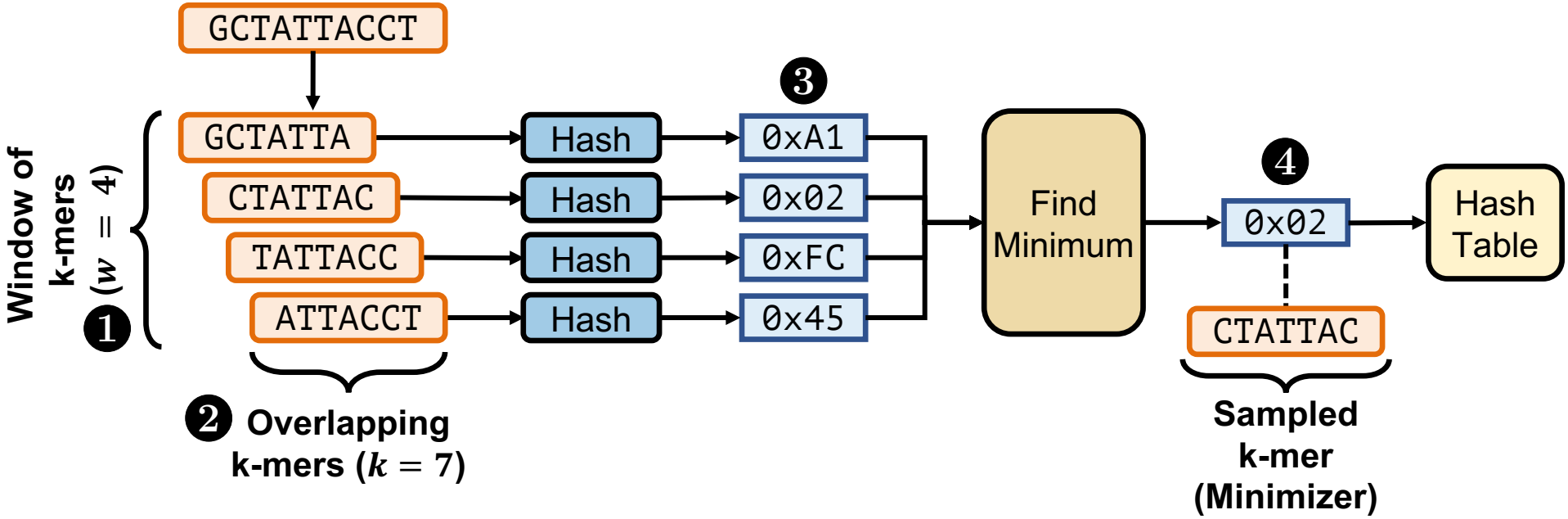
SAFARI

Backup Slides

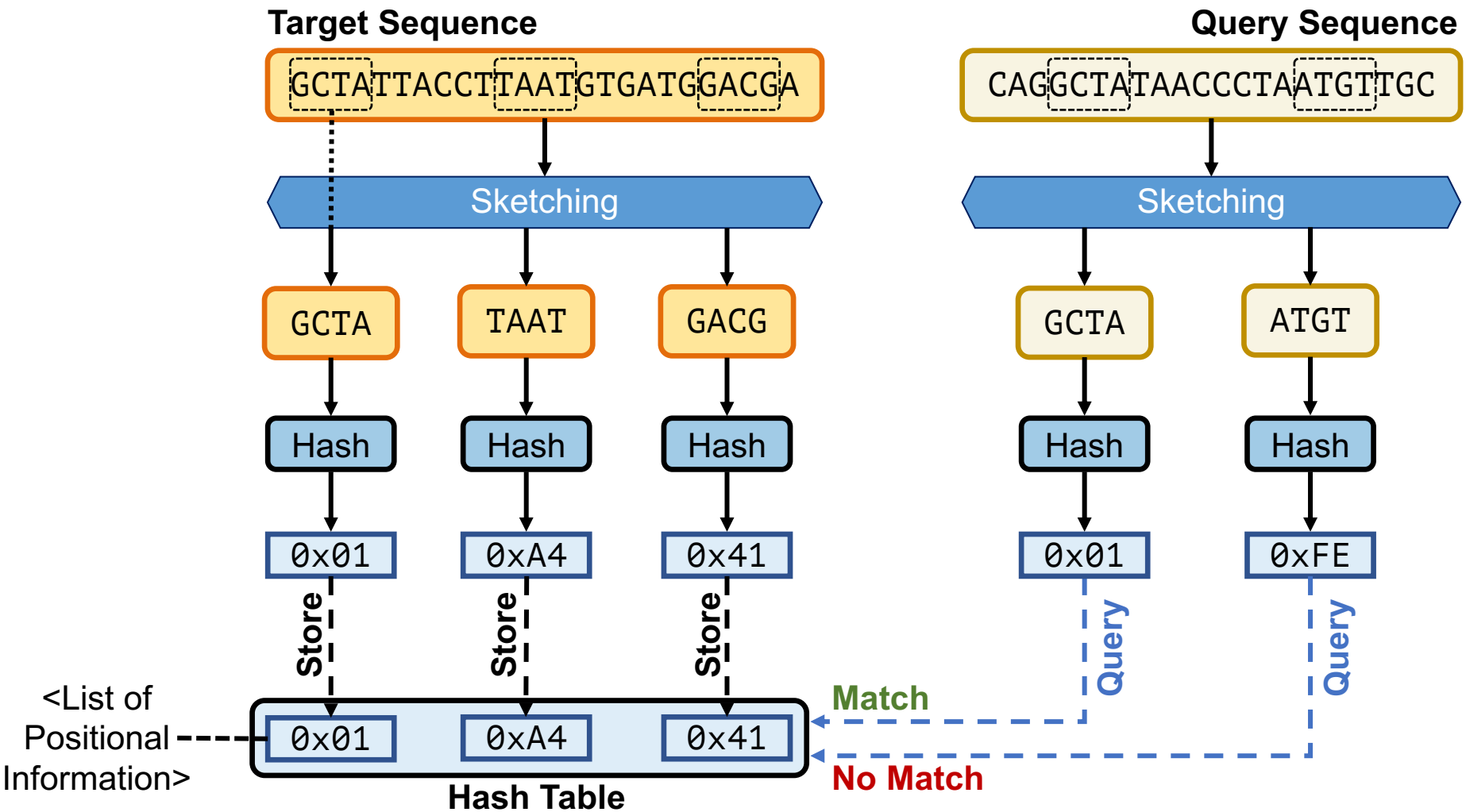
A Common Genome Analysis Pipeline



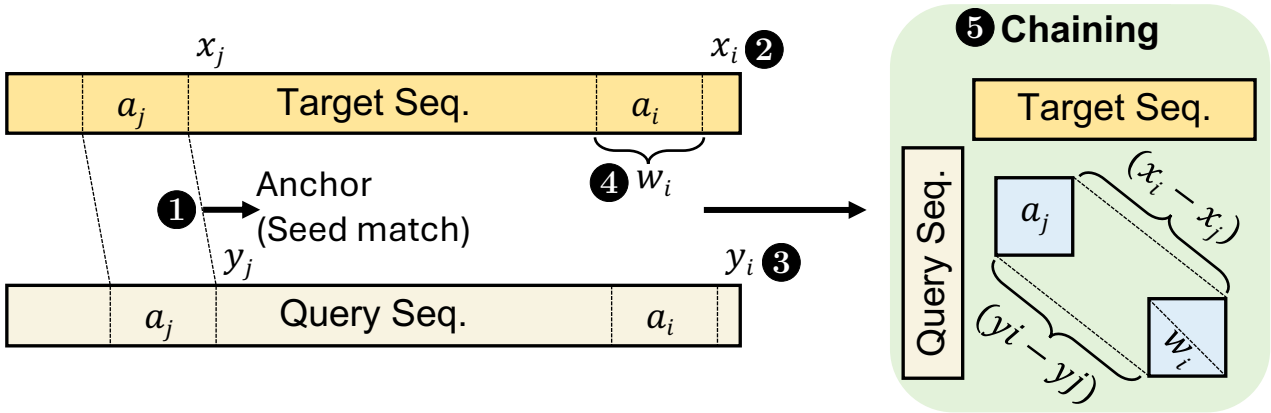
Minimizer Sketching



Hash-Based Sketching and Seed Matching



Chaining (Two Points)

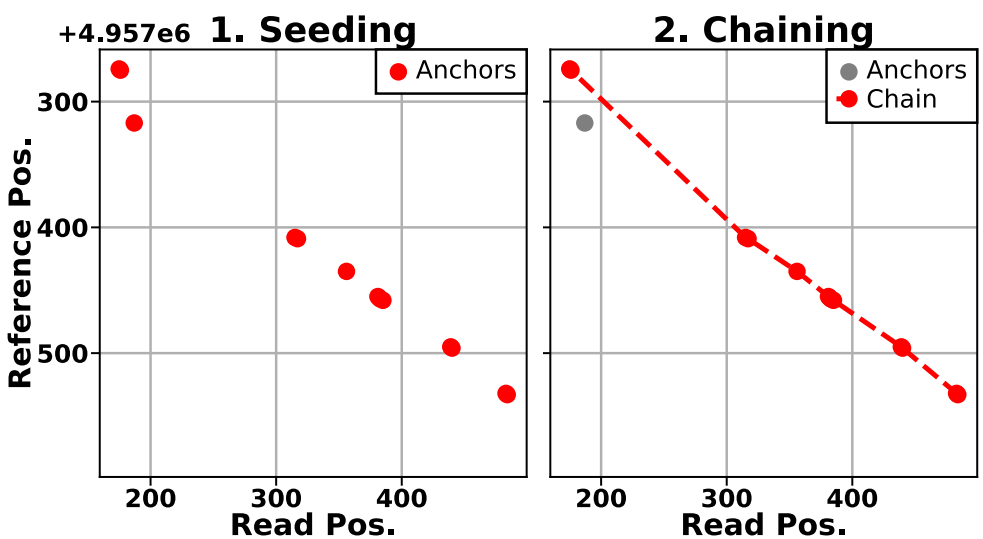


Chaining (Multiple Points)

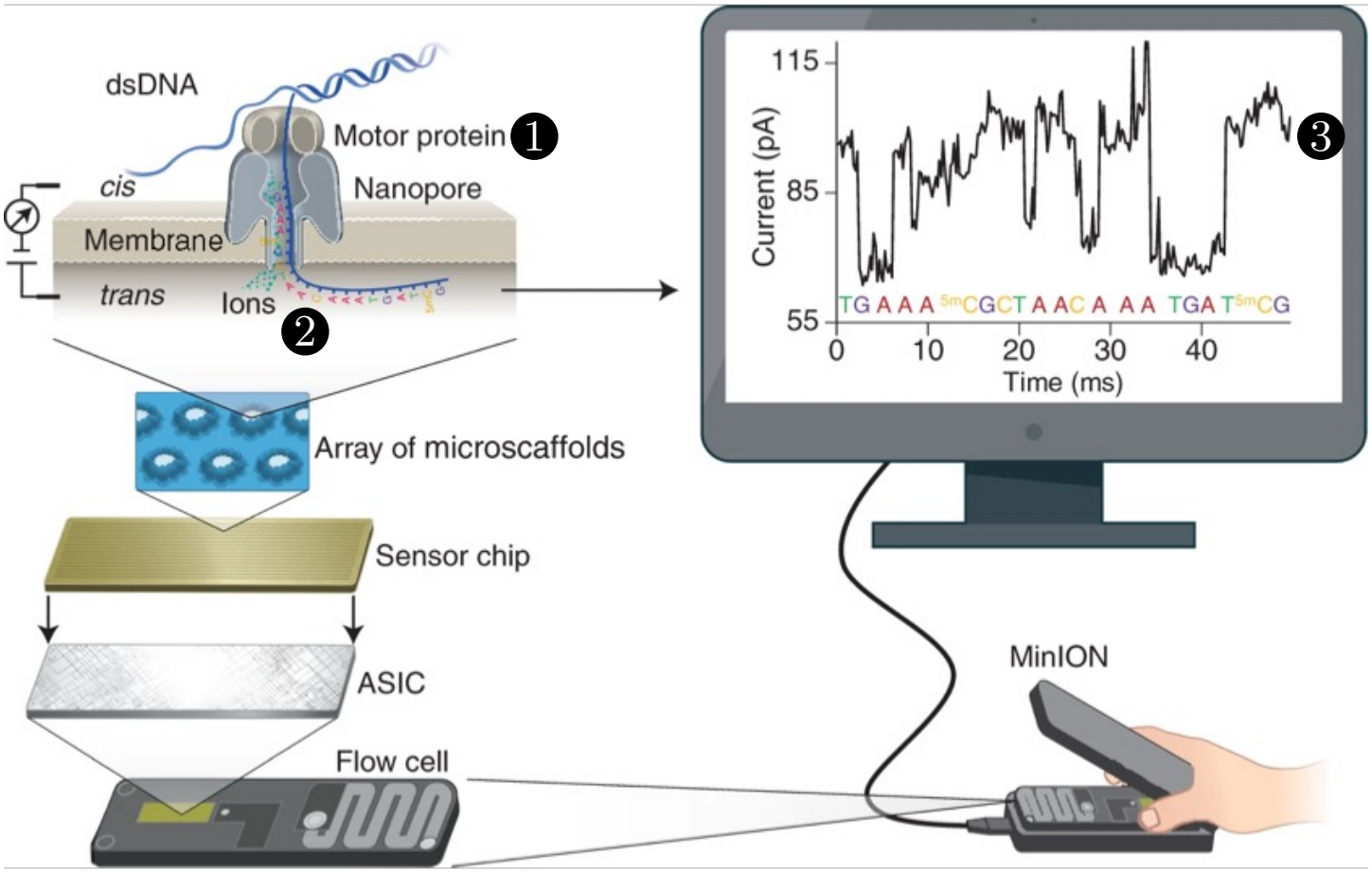
- **Exact hash value matches:** Needed for finding matching regions between a reference genome and a read

- What if there are mutations or errors?
  - **No hash (seed) match** will occur in such positions

- The chaining algorithm links **exact matches in a proximity** even though there are gaps (no seed matches) between them



Nanopore Sequencing



Source of Noise in Nanopore Sequencing

- **Stochastic thermal fluctuations in the ionic current**
  - Random ionic movement due to inherent thermal energy (Brownian motion)

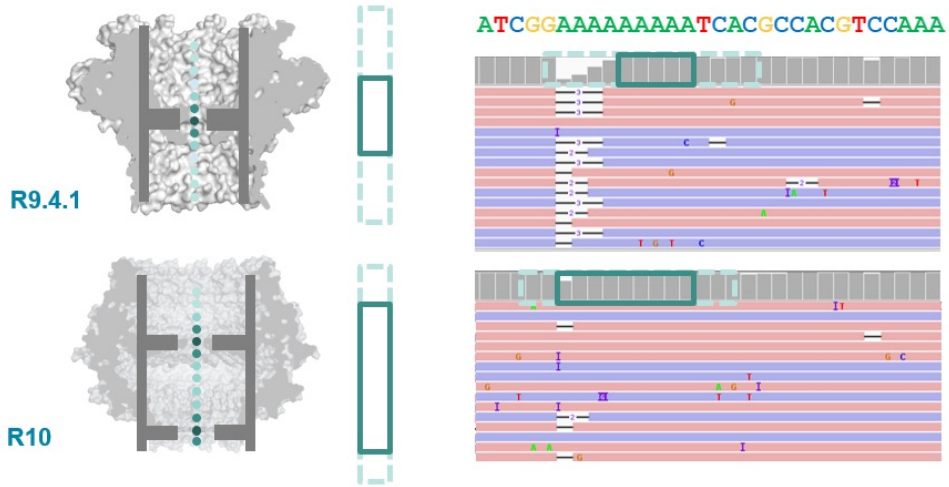
- **Variations in the translocation speed**
  - Mainly due to the motor protein

- **Environmental factors**
  - **Temperature:** Affecting enzymes including the motor protein
  - **pH levels:** Affecting charge and the shape of molecules

- **Maybe: Aging & material-related noise between nanopores**
  - Their effects potentially can be minimized with normalization techniques

R9 vs. R10 Chemistries

- **Dual reader head**



- **Motor protein** with more consistent translocation speed in R10

- **Duplex sequencing** in R10

Challenges in Real-Time Analysis

Rapid analysis to match the nanopore sequencer throughput

Timely decisions to stop sequencing as early as possible

Accurate analysis from noisy raw signal data

Power-efficient computation for scalability and portability

Applications of Read Until

Depletion: Reads mapping to a particular reference genome is ejected

- Microbiome studies by removing host DNA
- Eliminating known residual DNA or RNA (e.g., mitochondrial DNA)
- High abundance genome removal

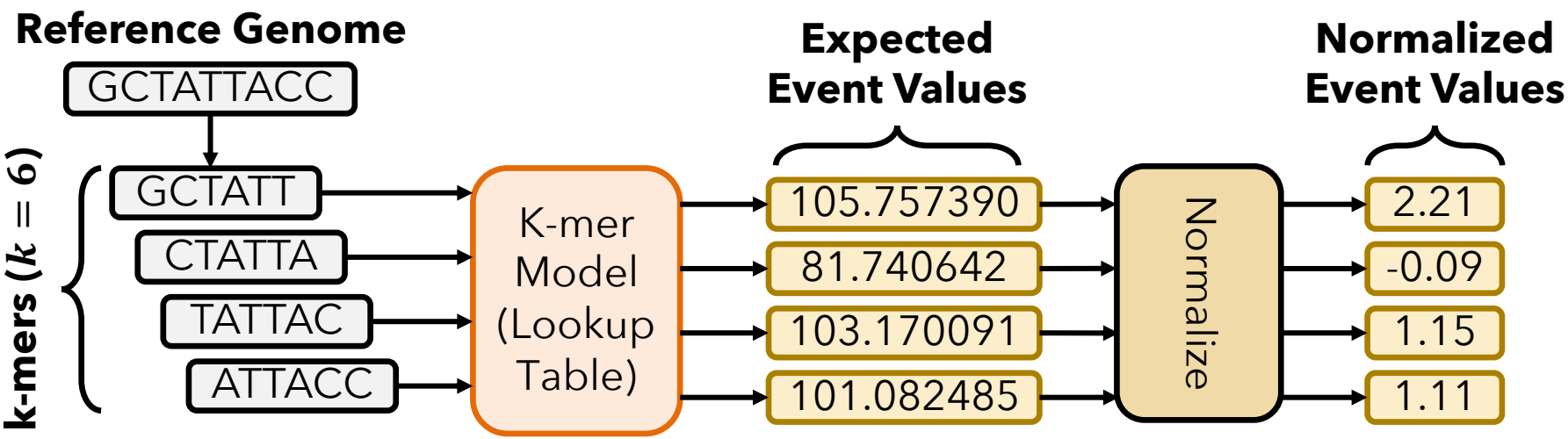
Enrichment: Reads **not** mapping to a particular reference genome is ejected

- Removing contaminated organisms
- Targeted sequencing (e.g., to a particular region of interest in the genome)
- Low abundance genome enrichment

Reference-to-Event Conversion

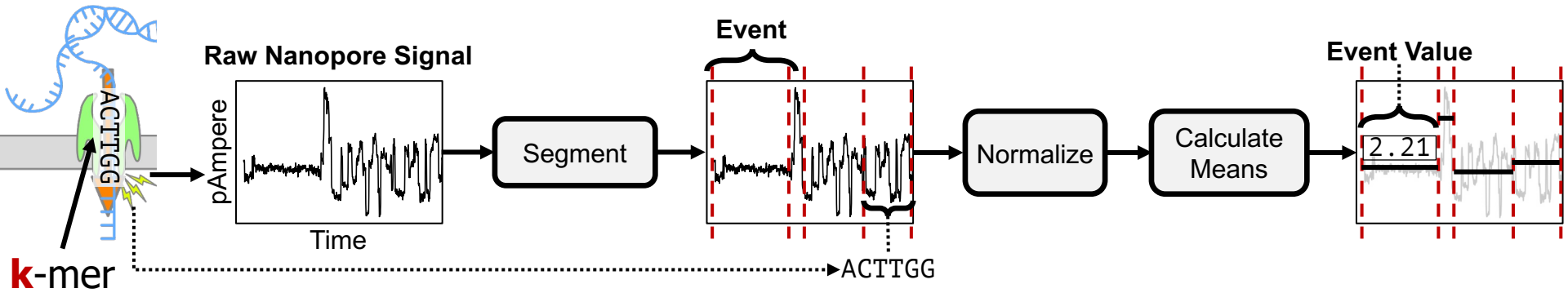
- **K-mer model:** Provides **expected** event values **for each k-mer**
  - Preconstructed based on nanopore sequencer characteristics

- Use the **k-mer model** to convert **all k-mers** of a reference genome to their **expected** event values



Enabling Analysis From Electrical Signals

- K many nucleotides (k-mers) sequenced at a time
- **Event:** A **segment** of the raw signal
  - Corresponds to a **particular** k-mer



- **Observation:** Event values generated after sequencing **the same k-mer** are **similar** in value (not necessarily the same)

Datasets

	Organism	Device Type	Reads (#)	Bases (#)	Avg. Read Length	Estimated Coverage (×)	SRA Accession
D1	*SARS-CoV-2*	MinION	10,001	4.02M	402	135×	CADDE Centre
D2	*E. coli*	GridION	353,948	2,332M	6,588	445×	ERR9127551
D3	*Yeast*	MinION	50,023	385M	7,698	32×	SRR8648503
D4	*Green Algae*	PromethION	30,012	622M	20,731	5.6×	ERR3237140
D5	*Human*	MinION	270,006	1,773M	6,567	0.6×	FAB42260

Throughput

	D1 *SARS-CoV-2*	D2 *E. coli*	D3 *Yeast*	D4 *Green Algae*	D5 *Human*
Throughput	2,065,764	2,720,702	2,128,800	1,668,065	3,579,472

Performance

Organism	Tool	Elapsed time (hh:mm:ss)	CPU time (sec)	Peak Mem. (GB)
D1	**Rawsamble**	0:00:03	33	1.07
SARS-CoV-2	**Minimap2**	0:00:01 (0.33×)	19 (0.58×)	0.16 (0.15×)
	Minimap2 + Dorado CPU (Fast)	0:01:45 (35.00×)	3,227 (97.79×)	44.93 (41.99×)
	Minimap2 + Dorado CPU (HAC)	0:05:45 (115.00×)	5,457 (165.36×)	57.98 (54.19×)
	Minimap2 + Dorado GPU (HAC)	0:01:41 (33.67×)	NA	0.8 (0.75×)
	Minimap2 + Dorado GPU (SUP)	0:25:47 (515.67×)	NA	1.23 (1.15×)
D2	**Rawsamble**	1:12:44	132,758	6.72
E. coli	**Minimap2**	0:14:25 (0.20×)	25,721 (0.19×)	26.73 (3.98×)
	Minimap2 + Dorado CPU (Fast)	7:17:05 (6.01×)	583,358 (4.39×)	50.43 (7.50×)
	Minimap2 + Dorado CPU (HAC)	32:26:12 (26.76×)	1,335,697 (10.06×)	38.0 (5.65×)
	Minimap2 + Dorado GPU (HAC)	0:36:14 (0.50×)	NA	26.73 (3.98×)
	Minimap2 + Dorado GPU (SUP)	1:30:30 (1.24×)	NA	26.73 (3.98×)
D3	**Rawsamble**	0:01:18	2,241	6.39
Yeast	**Minimap2**	0:00:21 (0.27×)	290 (0.13×)	5.25 (0.82×)
	Minimap2 + Dorado CPU (Fast)	0:54:04 (41.59×)	71,796 (32.04×)	56.13 (8.78×)
	Minimap2 + Dorado CPU (HAC)	3:13:56 (149.18×)	193,640 (86.41×)	65.43 (10.24×)
	Minimap2 + Dorado GPU (HAC)	0:04:33 (3.50×)	NA	5.25 (0.82×)
	Minimap2 + Dorado GPU (SUP)	0:10:33 (8.12×)	NA	5.92 (0.93×)
D4	**Rawsamble**	0:07:57	14,064	8.67
Green Algae	**Minimap2**	0:00:47 (0.10×)	882 (0.06×)	8.7 (1.00×)
	Minimap2 + Dorado CPU (Fast)	1:16:35 (9.63×)	79,606 (5.66×)	50.88 (5.87×)
	Minimap2 + Dorado CPU (HAC)	4:30:07 (33.98×)	286,362 (20.36×)	64.07 (7.39×)
	Minimap2 + Dorado GPU (HAC)	0:06:01 (0.76×)	NA	8.7 (1.00×)
	Minimap2 + Dorado GPU (SUP)	0:14:54 (1.87×)	NA	8.7 (1.00×)
D5	**Rawsamble**	0:28:56	51,975	6.0
Human	**Minimap2**	0:01:52 (0.06×)	1,372 (0.03×)	20.21 (3.37×)
	Minimap2 + Dorado CPU (Fast)	6:42:24 (13.91×)	802,983 (15.45×)	81.98 (13.66×)
	Minimap2 + Dorado CPU (HAC)	23:27:18 (48.64×)	1,219,043 (23.45×)	46.12 (7.69×)
	Minimap2 + Dorado GPU (HAC)	0:20:24 (0.71×)	NA	20.31 (3.38×)
	Minimap2 + Dorado GPU (SUP)	1:05:48 (2.27×)	NA	20.21 (3.37×)

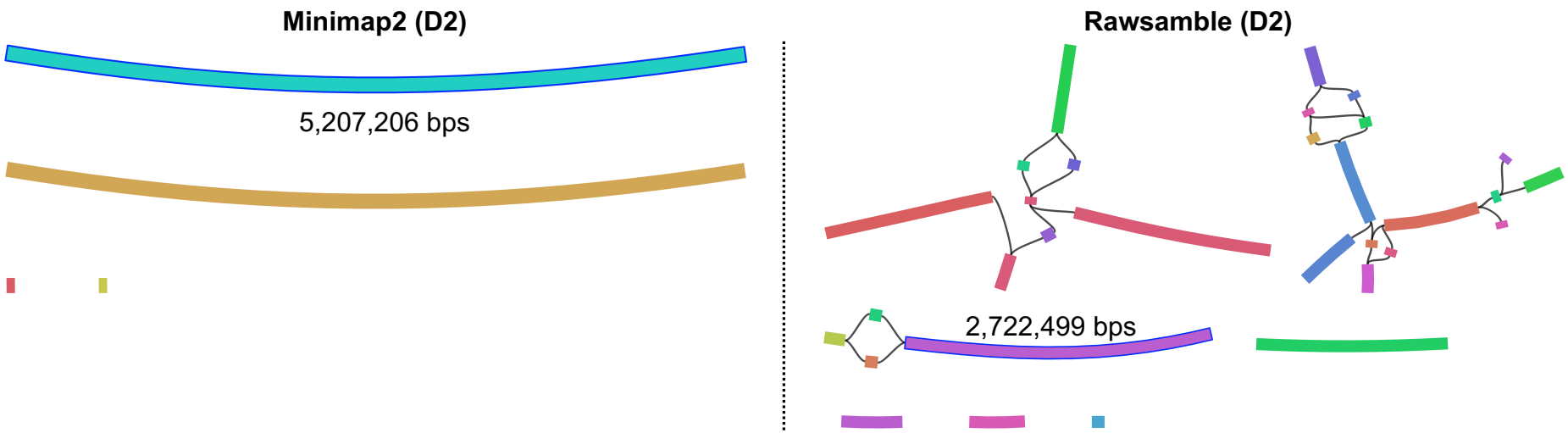
Overlapping Statistics

	Organism	Unique to Rawsamble (%)	Unique to Minimap2 (%)	Shared Overlaps (%)
D1	*SARS-CoV-2*	11.55	15.27	73.18
D2	*E. coli*	8.33	50.62	41.05
D3	*Yeast*	24.94	35.17	39.89
D4	*Green Algae*	3.76	78.64	17.61
D5	*Human*	32.69	56.18	11.13

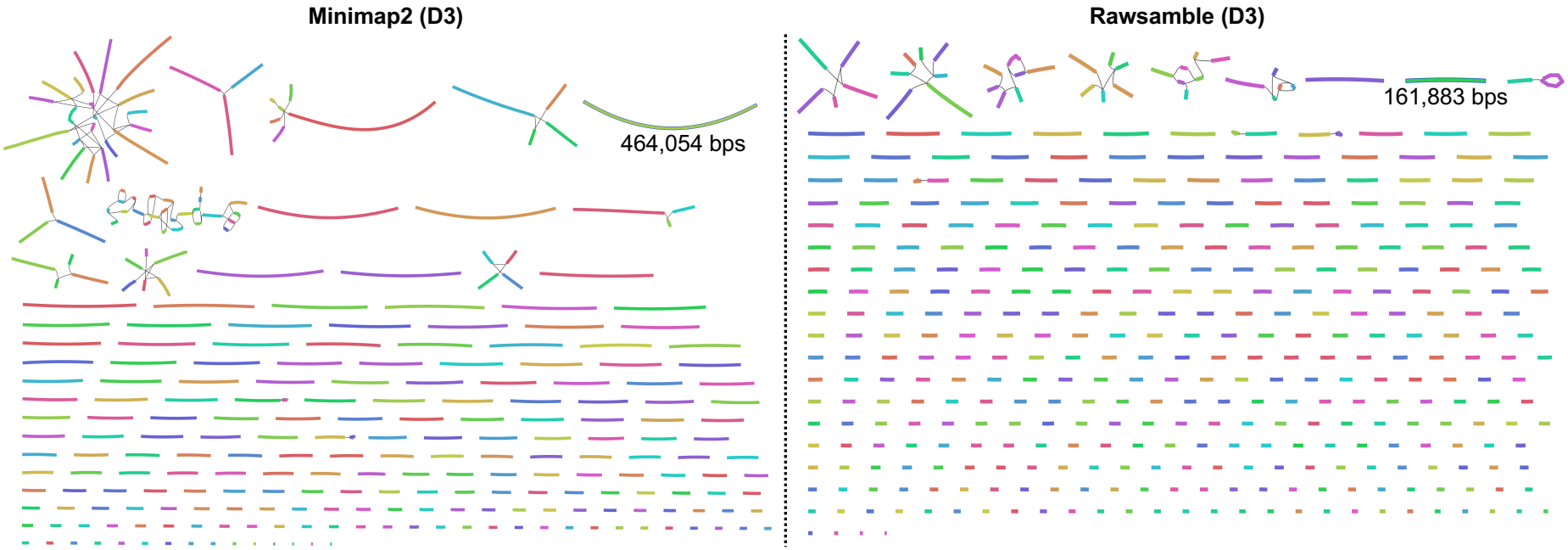
Assembly Statistics

Dataset	Tool	Total Length (bp)	Largest Comp. (bp)	N50 (bp)	auN (bp)	Longest Unitig (bp)	Unitig Count
D2	Rawsamble	14,525,505	4,841,669	1,535,079	1,309,738	2,722,499	31
E. coli	minimap2	10,434,542	5,207,206	5,204,754	5,194,738	5,207,206	4
	Gold standard	5,235,343	5,235,343	5,235,343	5,235,343	5,235,343	1
D3	Rawsamble	13,898,208	362,050	41,118	48,106	161,883	396
Yeast	minimap2	23,755,455	1,611,876	134,050	150,908	464,054	282
	Gold standard	11,963,521	11,835,059	640,934	623,210	1,073,346	68
D4	Rawsamble	3,448,899	448,422	93,111	108,818	252,038	50
Green Algae	minimap2	2,117,190	198,709	63,310	88,906	198,709	55
	Gold standard	106,479,288	2,255,807	452,774	538,136	1,667,975	420
D5	Rawsamble	1,850,419	493,004	51,300	116,049	364,113	48
Human	minimap2	747,607	65,951	19,476	22,103	48,424	61
	Gold standard	8,365,210	367,305	19,329	29,697	150,470	592

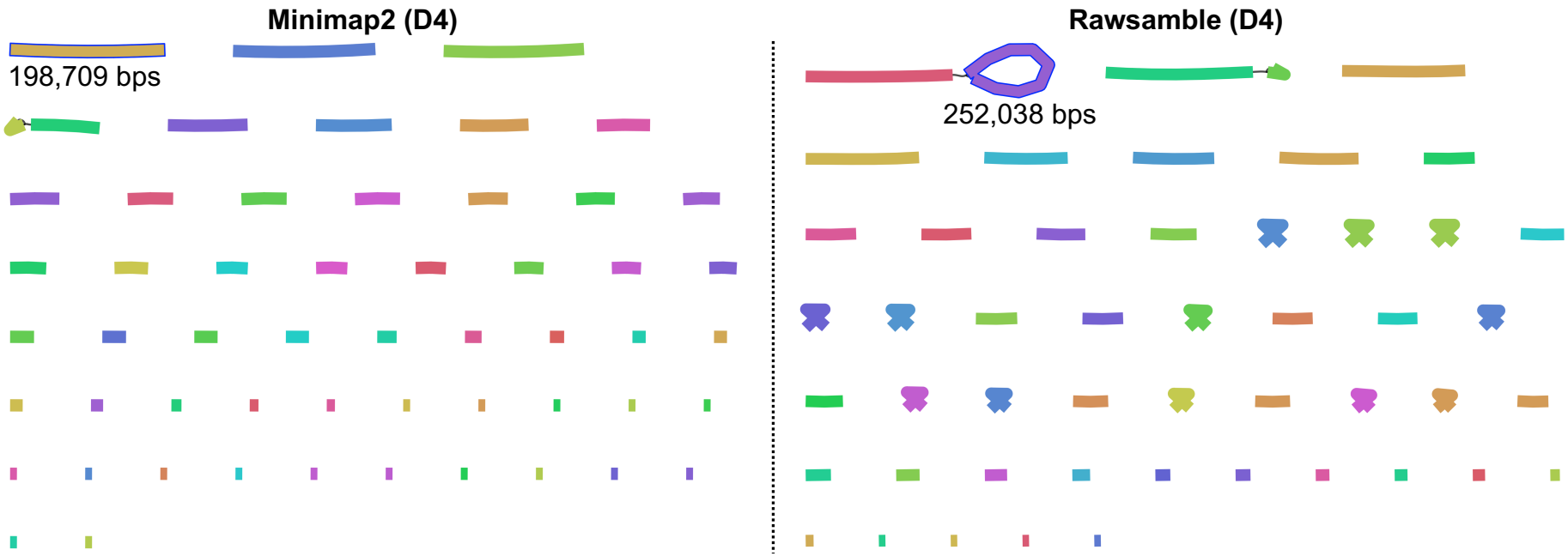
Visualizing the E. coli Assembly Graph



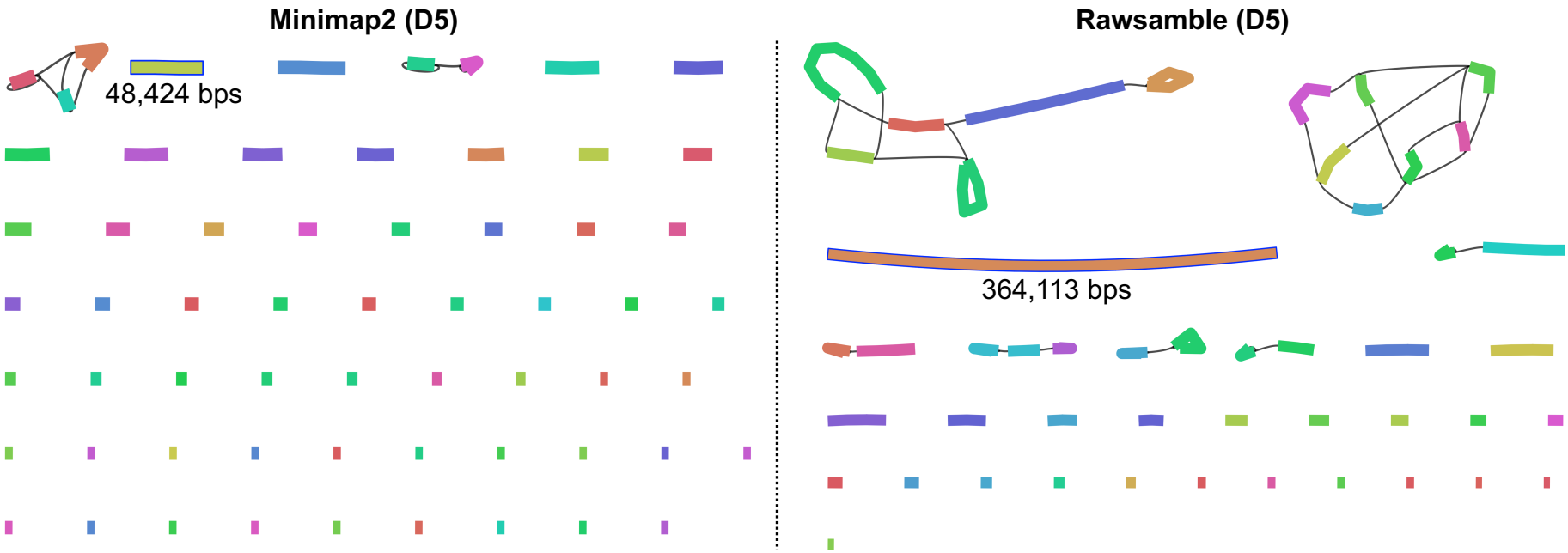
Visualizing the Yeast Assembly Graph



Visualizing the Green Algae Assembly Graph



Visualizing the Human Assembly Graph



HERRO Correction Before and After

Dataset	Coverage Before Correction	Coverage After Correction
D2 *E. coli*	445×	240×
D3 *Yeast*	32×	12×
D4 *Green algae*	5.6×	3.7×
D5 *Human*	0.6×	0.002×

Parameters

Tool	D1 *SARS-CoV-2*	D2 *E. coli*	D3 *Yeast*	D4 *Green Algae*	D5 *Human*
Rawsamble	-x ava-viral -t 32	-x ava -t 32	-x ava -t 32	-x ava -t 32	-x ava –chain-gap-scale 0.6 -t 32
Minimap2	-x ava-ont –for-only -t 32				
Dorado CPU (Fast)	basecaller -x cpu dna_r9.4.1_e8_fast@v3.4				
Dorado CPU (HAC)	basecaller -x cpu dna_r9.4.1_e8_hac@v3.3				
Dorado GPU (HAC)	basecaller dna_r9.4.1_e8_hac@v3.3				
Dorado GPU (SUP)	basecaller dna_r9.4.1_e8_sup@v3.3				
Miniasm					

Presets

Preset	Corresponding parameters	Usage
ava-viral	-e 6 -q 4 -w 0 –sig-diff 0.45 –fine-range 0.4 –min-score 20 –min-score2 30 –min-anchors 5 –min-mapq 5 –bw 1000 –max-target-gap 2500 –max-query-gap 2500 –chain-gap-scale 1.2 –chain-skip-scale 0.3	Viral genomes
ava	-e 8 -q 4 -w 3 –sig-diff 0.45 –fine-range 0.4 –min-score 40 –min-score2 75 –min-anchors 5 –min-mapq 5 –bw 5000 –max-target-gap 2500 –max-query-gap 2500	Default case

Versions

Tool	Version
Rawsamble	2.1
Minimap2	2.24
Dorado	0.7.3
Miniasm	0.3-r179
Rawasm	main
Flye	2.9.5
HERRO	0.1

Future Work

Reverse Complementing Raw Nanopore Signals

- Without reverse complementing, we are missing half of the useful information

Dynamically Building the Hash Table in Real-Time

- Needed for real-time *de novo* assembly construction
- What are the useful applications for real-time *de novo* assembly construction?