A Lightweight Algorithm-Architecture Co-Design for Mitigating the Data Preparation Bottleneck in Large-Scale Genome Sequence Analysis

Nika Mansouri Ghiasi

Talu Güloglu Harun Mustafa Can Firtina Konstantina Koliogeorgi

Konstantinos Kanellopoulos Haiyu Mao Rakesh Nadig

Mohammad Sadrosadati Jisung Park Onur Mutlu

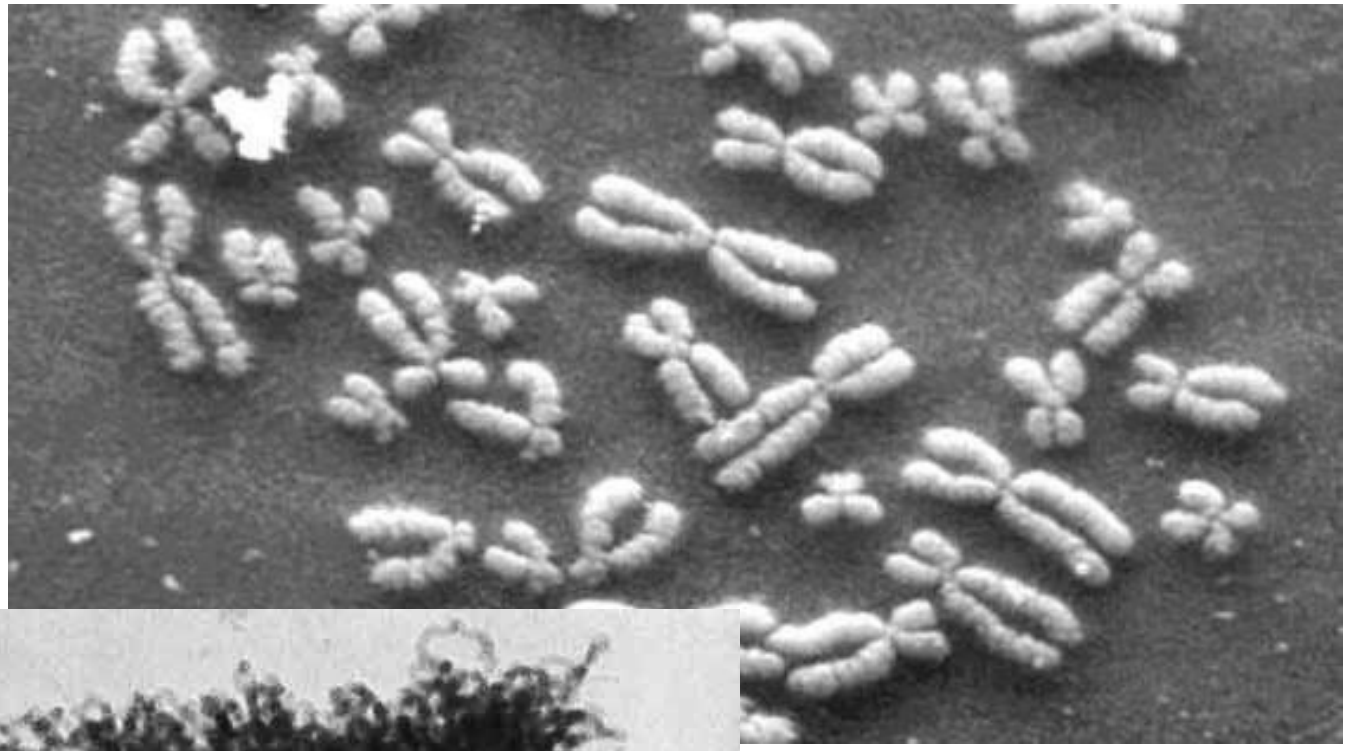
SAFARI

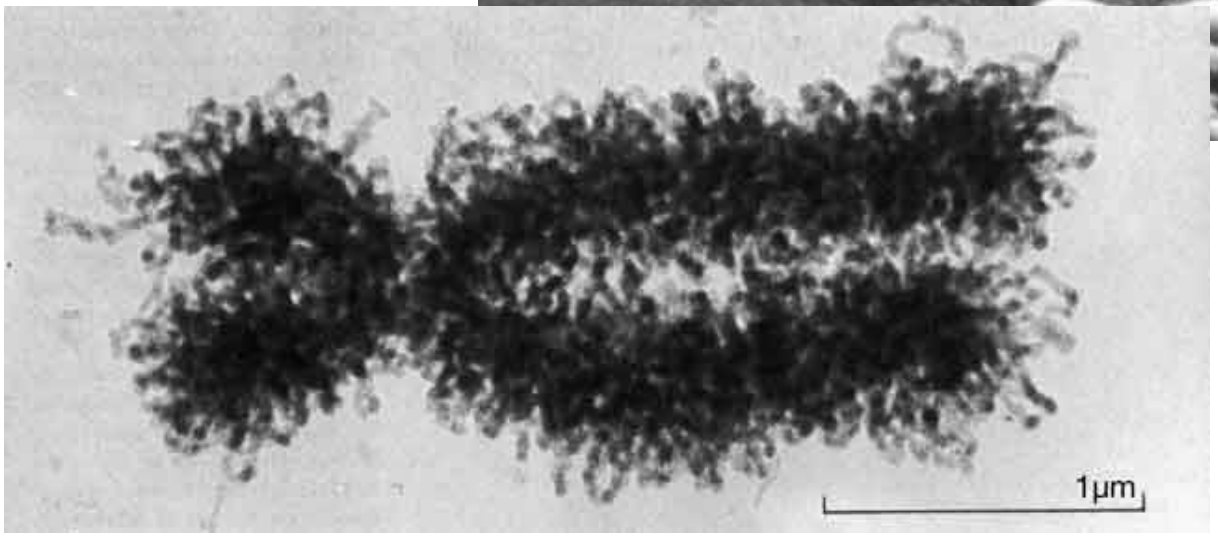
ETH zürich UNIVERSITY OF MARYLAND KING'S College LONDON POSTECH

Applications of Genome Sequence Analysis

& Many More…

DNA Under Electron Microscope





human chromosome #12
from a HeLa cell

Digital Format for Analysis

CCTCCTCAGTGCCACCCAGCCCACTGGCAGCTCCCA
AACAGGCTCTTATTAAAACACCCTGTTCCCTGCCCC
TTGGAGTGAGGTGTCAAGGACCTAAACTAAAAAAAA
AAAAAGAAAAAGAAAAGAAAAAGAATTTAAAATTTA
AGTAATTCTTTGAAAAAAACTAATTTCTAAGCTTCT
TCATGTCAAGGACCTAATGTGCTAAACAGCACTTTT
TTGACCATTATTTTGGATCTGAAAGAAATCAAGAAT
AAATGAAGGACTTGATACATTGGAAGAGGAGAGTCA
AGGACCTACAGAAAAAAAAAAAAAAGAAAAAGAAAA
GAAAAAGAATTTAAAATTTAAGTAATTCTTTGAAAA
AAACTAATTTCTAAGCTTCTTATGTCAAGGACCTAA
TGTCTGTGTTGCAGGTCTTCTTGCATCTTCCCTGTC

High-Throughput Sequencers

Illumina MiSeq

Pacific Biosciences Sequel II

Oxford Nanopore PromethION

Oxford Nanopore MinION

Illumina NovaSeq 6000

Pacific Biosciences RS II

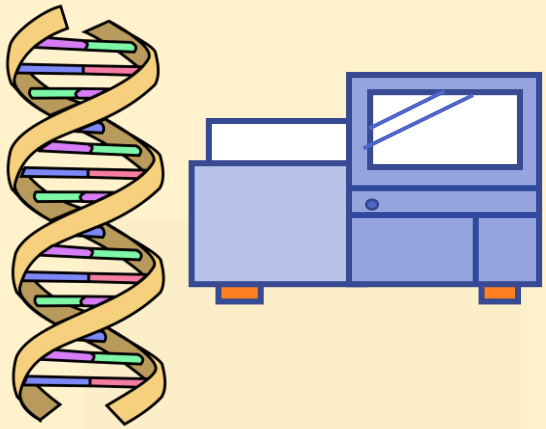
Oxford Nanopore GridION

… and more! All produce data with different properties.

High-Throughput Sequencers

Current sequencing technologies cannot sequence very long DNA molecules end-to-end

Instead, they extract smaller fragments of the original DNA sequence, known as reads

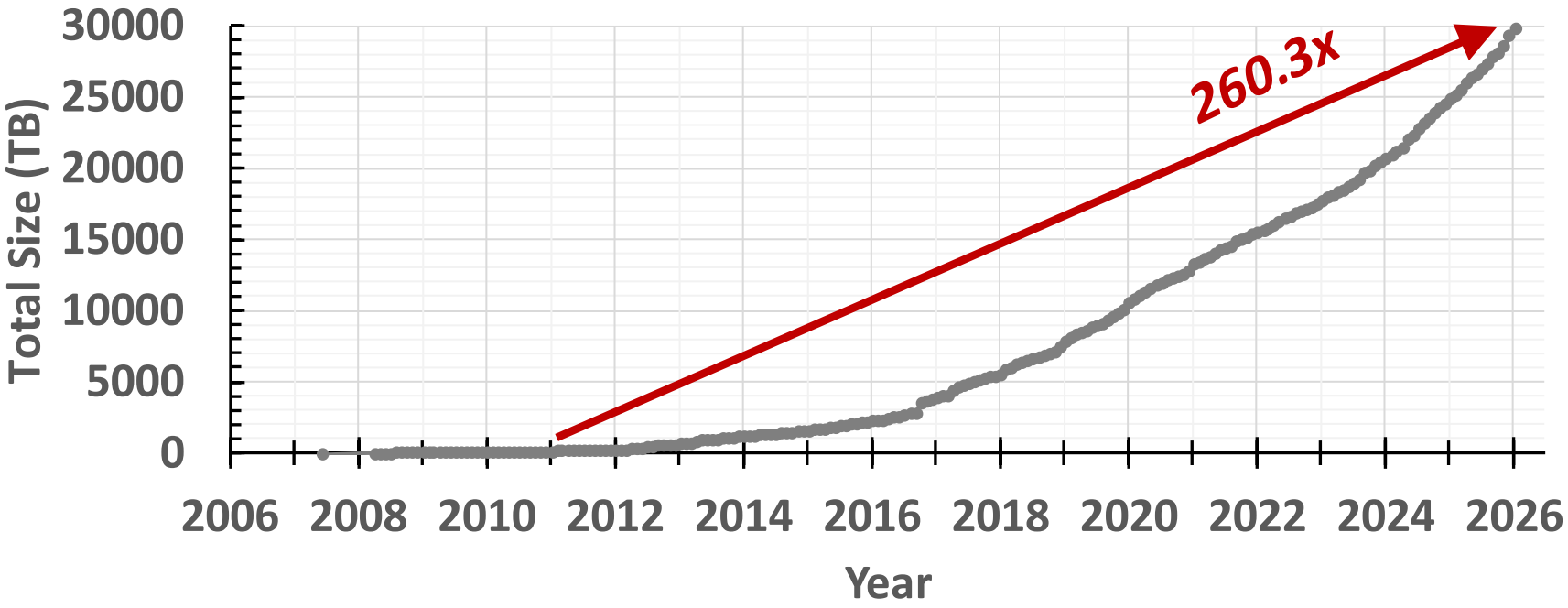


Genomic Data is Growing at a Fast Pace

Important Role in Many Fields

Significant Drop in Sequencing Cost

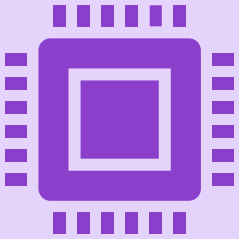
Fast growth in genomic data size



Public Genomic Sequence Reads Data Size in Sequence Read Archive (SRA) Over Years

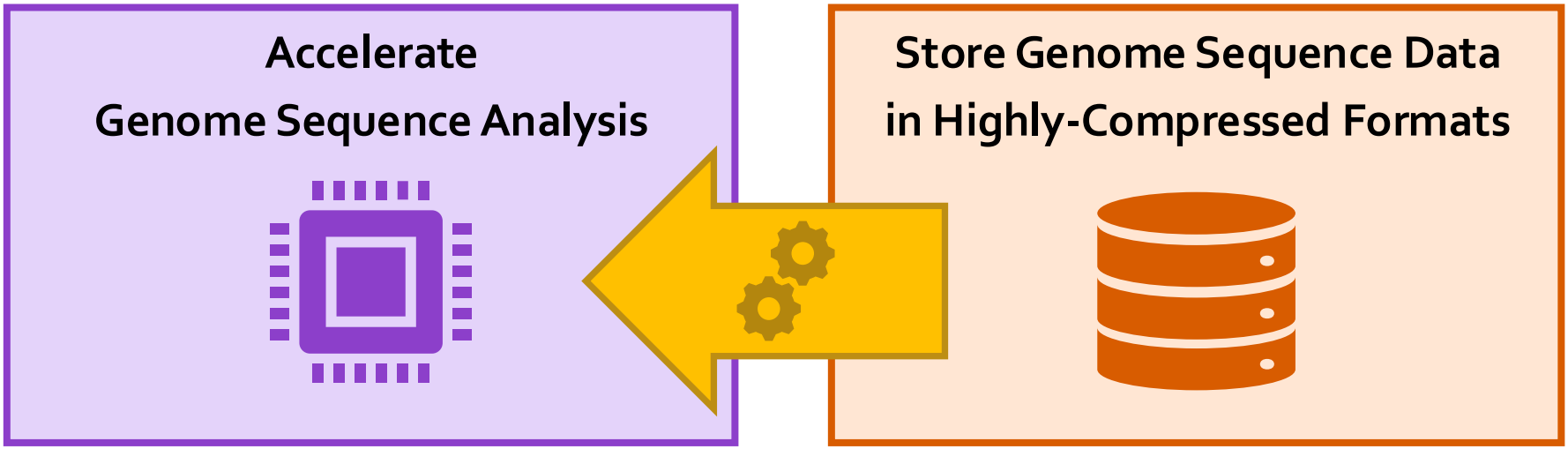
Large Efforts in Handling Fast-Growing Genomic Data

Accelerate
Genome Sequence Analysis



Store Genome Sequence Data
in Highly-Compressed Formats

Data Preparation Bottleneck

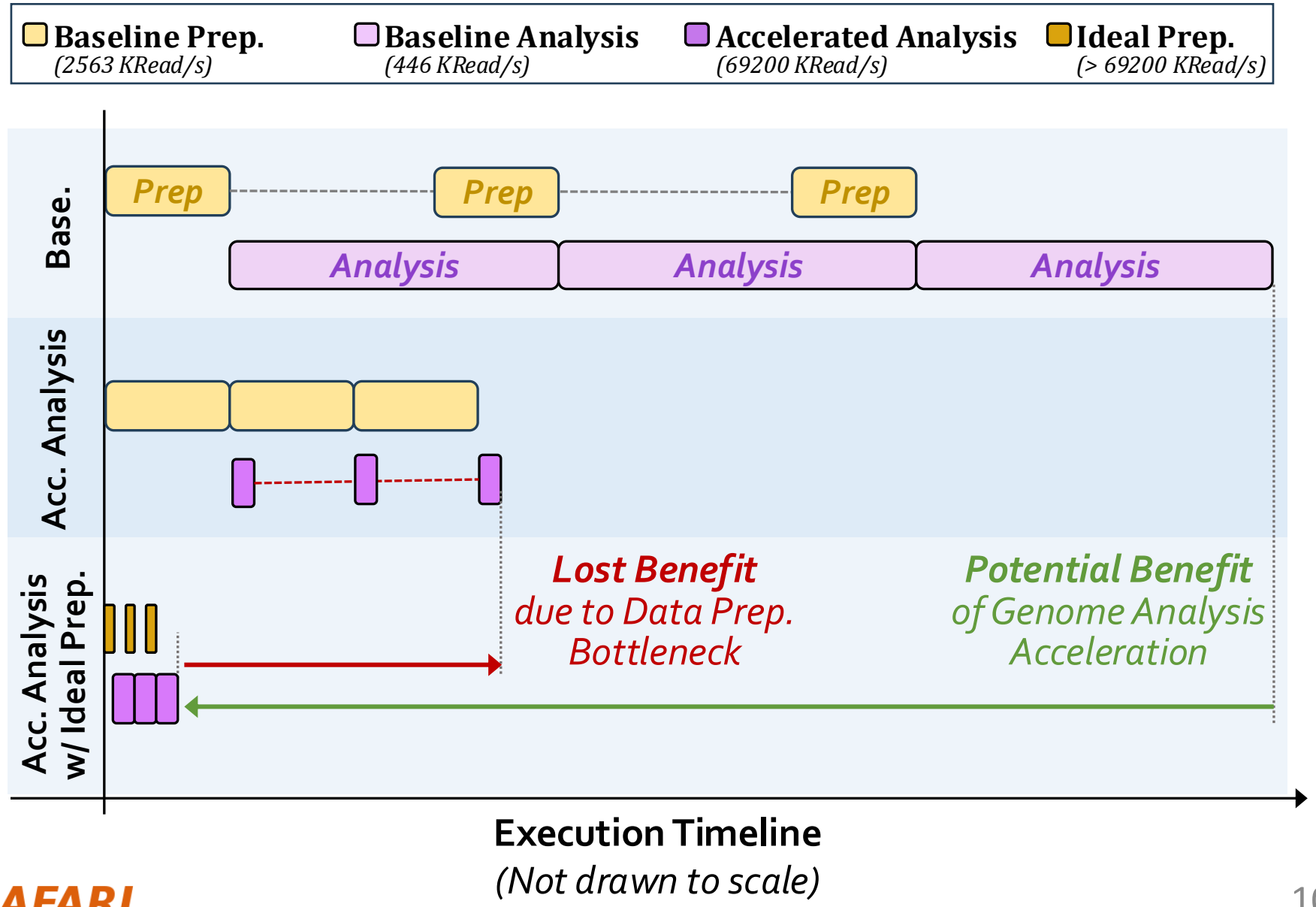


Genomic data is

decompressed *and* ***formatted*** *first*

before an accelerator can operate on it

Effect of the Data Preparation Bottleneck



Mitigating Data Preparation Bottleneck

We need to effectively mitigate the data preparation bottleneck while meeting four key requirements:

High performance

High energy efficiency

High compression ratio

Low overhead

Meeting all these requirements is challenging

Even more challenging in genome analysis systems used in hardware resource-constrained environments

SAGe

*An algorithm-architecture co-design for highly-compressed **s**torage and high-performance **a**ccess of large-scale **ge**nomic sequence data*

- **Key insight**: information encoded in genomic data follows specific properties
- We leverage these properties to co-design
  - A lossless (de)compression algorithm
  - Hardware for decompression with lightweight operations
  - Storage data layout & Interface commands to access data

Higher performance *(by 3.0×–32.1×)*

Higher energy efficiency *(by 13.0×–34.0x)*

High compression ratio

Low overhead

Outline

- Background
- Motivation and Goal
- SAGe
- Evaluation
- Conclusion

Genomics-Specific Compression

Read Set *(A set of reads from a sample)*

ATACGTAGAAAAAGT
TCGATGCTTGCATAA
AGCAAATGTACGATG
CATAAGTCGATAGTT
AAAAAGTCGATGCTT
...

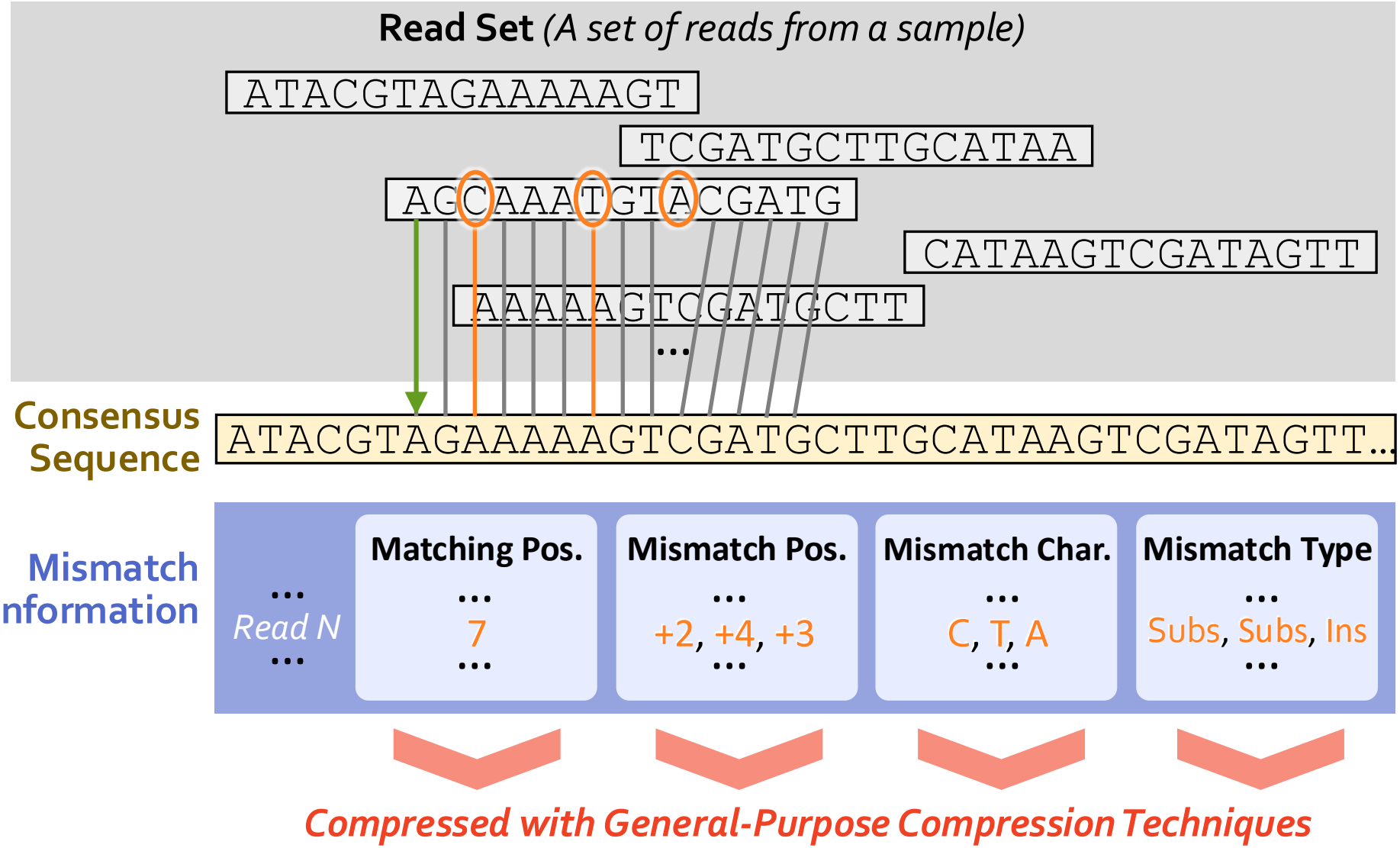
Consensus Sequence

ATACGTAGAAAAAGTCGATGCTTGCATAAGTCGATAGTT...

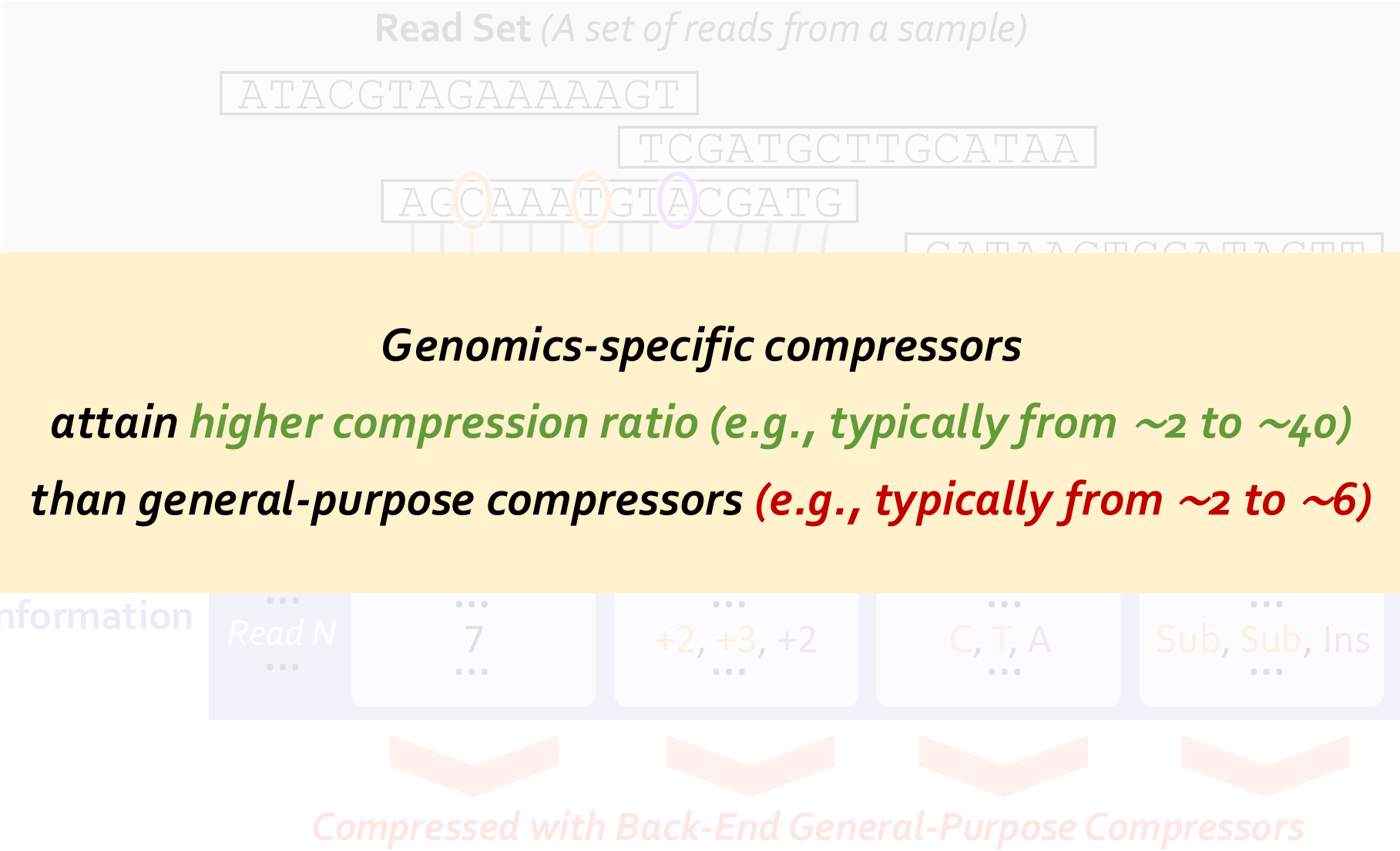
Mismatch Information

Genomics-Specific Compression

Genomics-Specific Compression



Genomics-Specific Compression



Genomics-specific compressors

attain higher compression ratio (e.g., typically from ~2 to ~40)

than general-purpose compressors (e.g., typically from ~2 to ~6)

Outline

Background

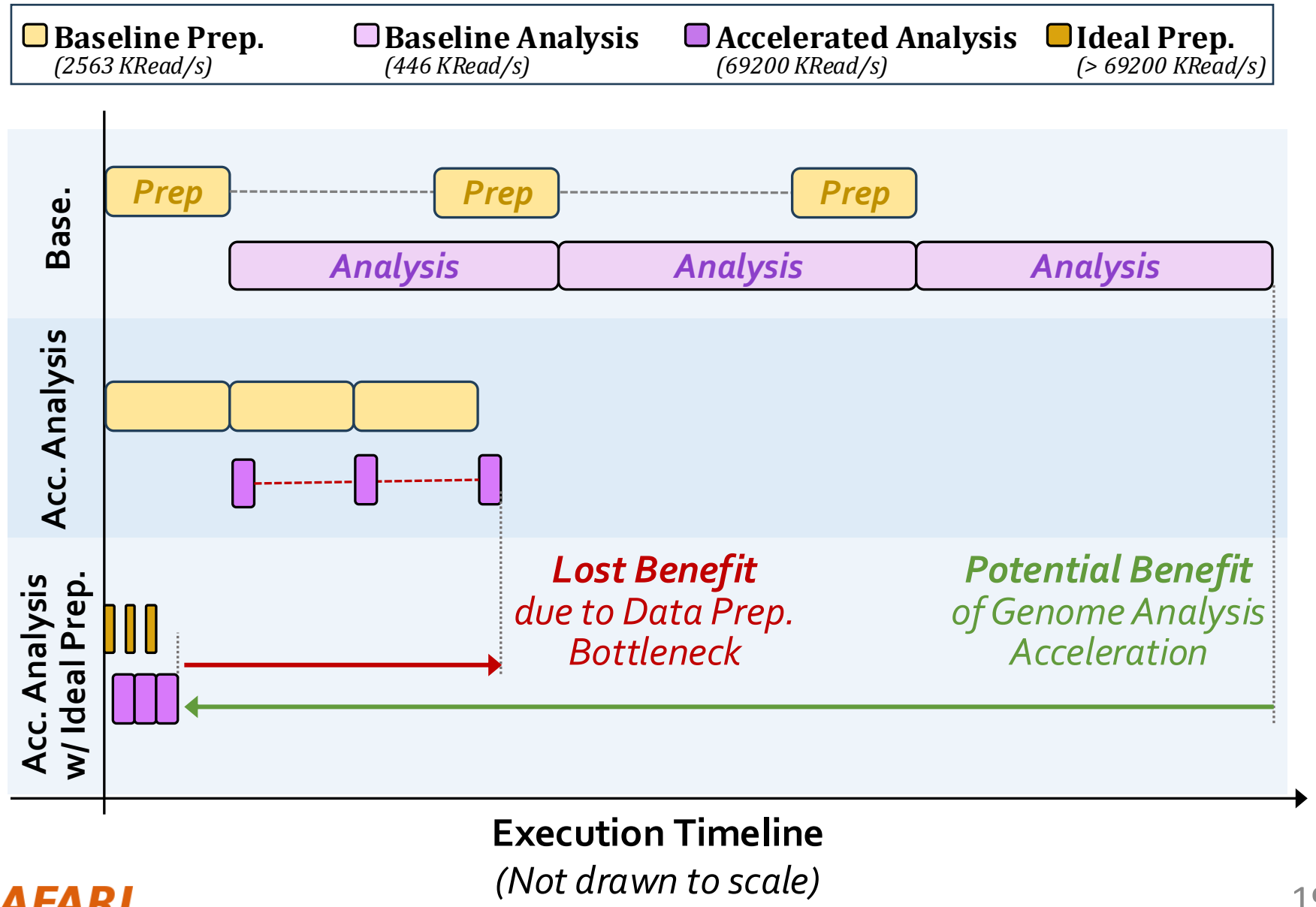
Motivation and Goal

SAGe

Evaluation

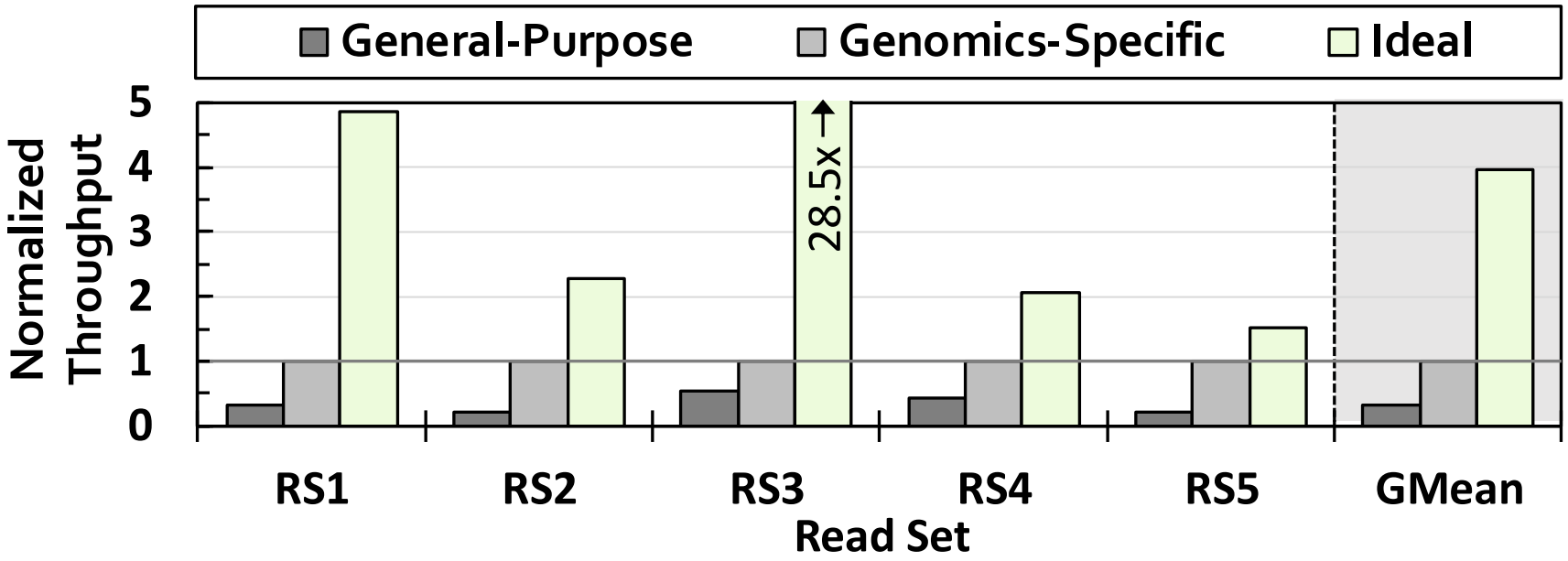
Conclusion

Effect of the Data Preparation Bottleneck



Data Preparation Bottleneck

- End-to-end performance of a genome analysis application, which includes both **data preparation** and **genome analysis**
  - Genome Analysis: GEM *[Chen+, IEEE TPDS'23]*, a state-of-the-art hardware accelerator for read mapping (a fundamental task in genome analysis)



Data preparation greatly limits end-to-end performance

Challenges

- Mitigating the data preparation bottleneck is challenging when aiming to achieve **high performance**, **energy efficiency**, **high compression ratio**, & **low overhead**

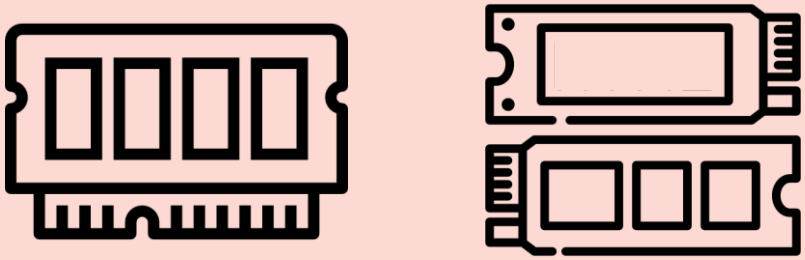
- This is because prior approaches rely on
  - **Expensive computational units**
  - **Large buffers, or large DRAM bandwidth or capacity**

- This challenge is exacerbated in genome analysis systems used in hardware **resource-constrained environments**

Portable Systems

Near-Data Processing Systems

(e.g., in Main Memory or Storage System)



Our Goal

Effectively mitigate the data preparation bottleneck, while meeting four key requirements:

High performance

Energy efficiency

High compression ratio

Low overhead

Outline

Background

Motivation and Gaol

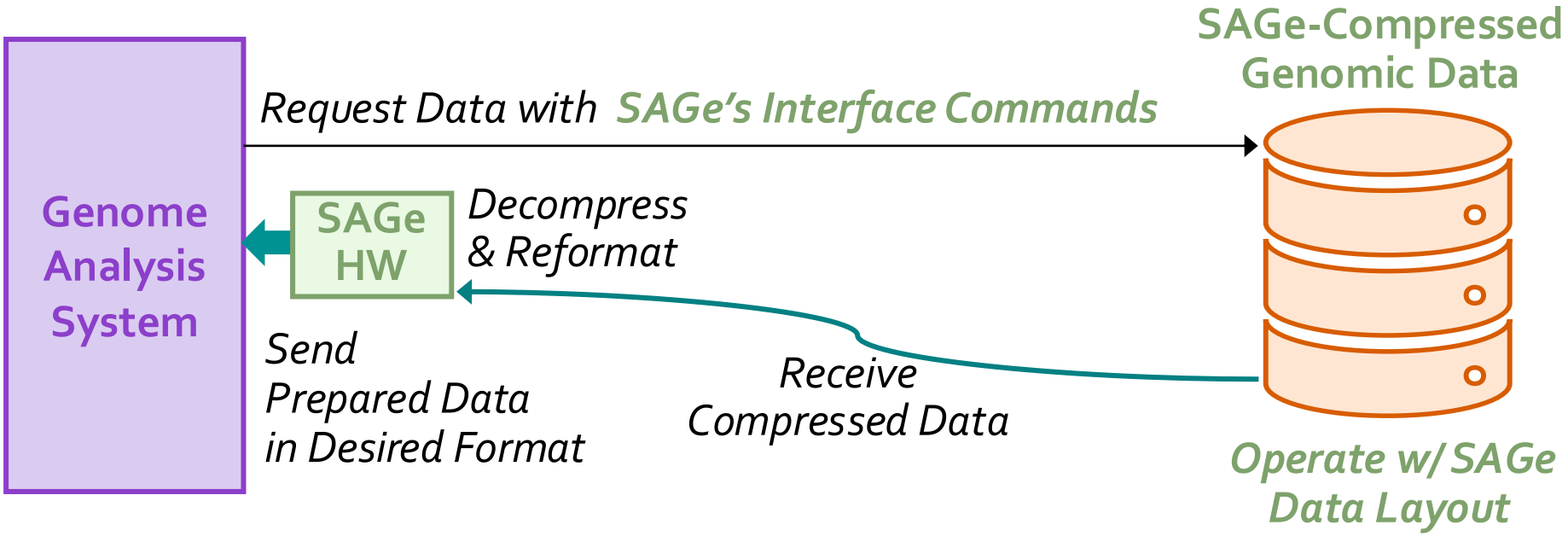
SAGe

Evaluation

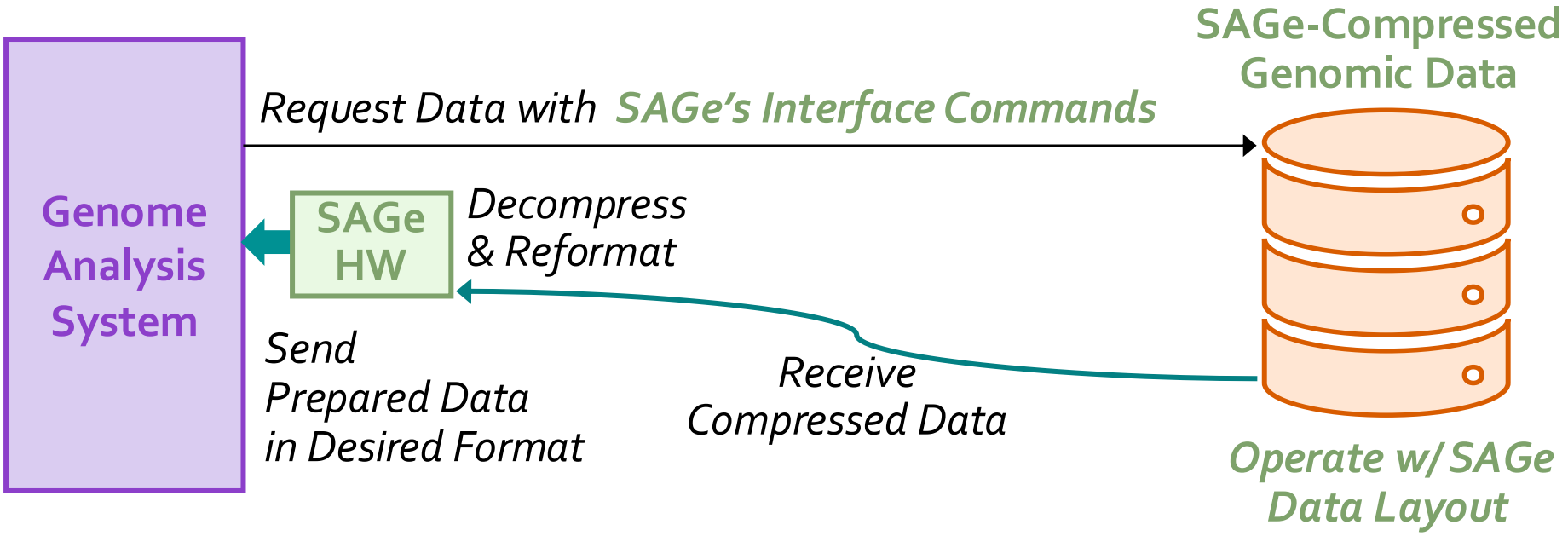
Conclusion

*An algorithm-architecture co-design for highly-compressed **s**torage and high-performance **a**ccess of large-scale **ge**nomic sequence data*

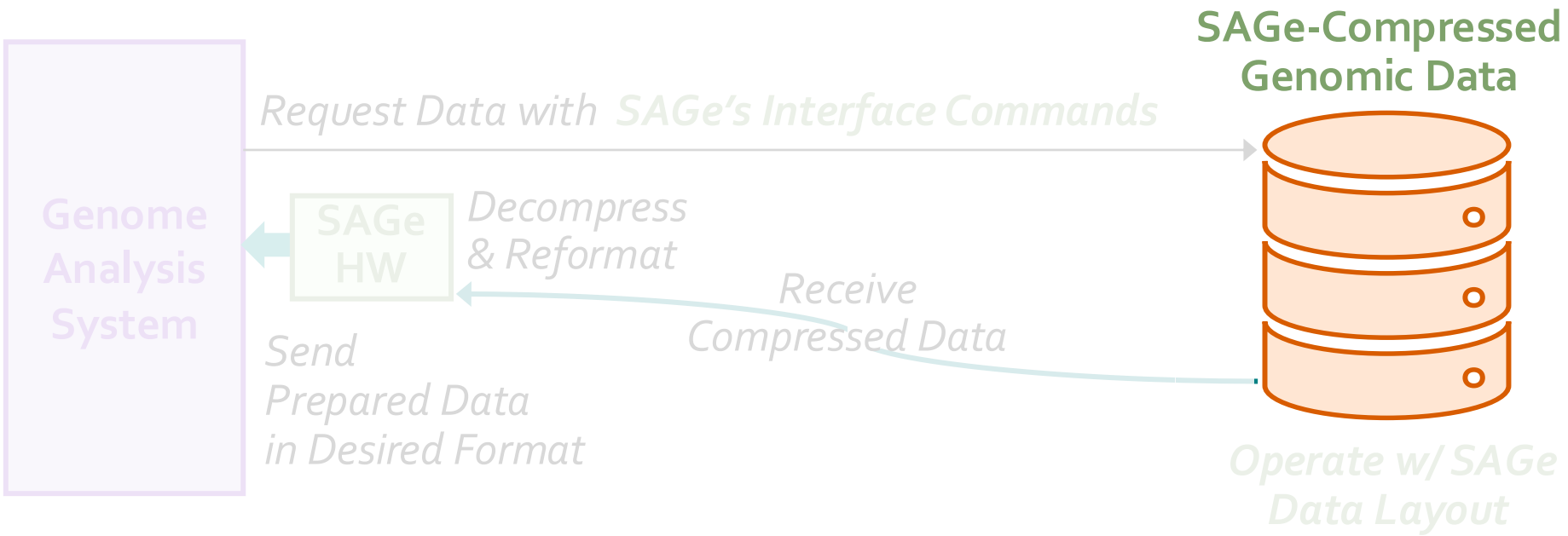
- **Key insight**: information encoded in genomic data follows specific properties
  We leverage these properties in our algorithm-architecture co-design



SAGe's Algorithm Architecture Co-Design



SAGe's Algorithm Architecture Co-Design



Compression Algorithm and Data Structures

Consensus Sequence

ATACGTAGAAAAAGTCGATGCTTGCATAAGTCGATAGTT...

Mismatch Information

	Matching Pos.	Mismatch Pos.	Mismatch Base	Mismatch Type
Read 1	7	1, 0, 9	C, T, A	Subs, Subs, Ins
Read 2	10	1, 1, 0	A, T, T	Ins, Subs, Ins
...	...	...	...	...

Compression Algorithm and Data Structures

Consensus Sequence

ATACGTAGAAAAAGTCGATGCTTGCATAAGTCGATAGTT...

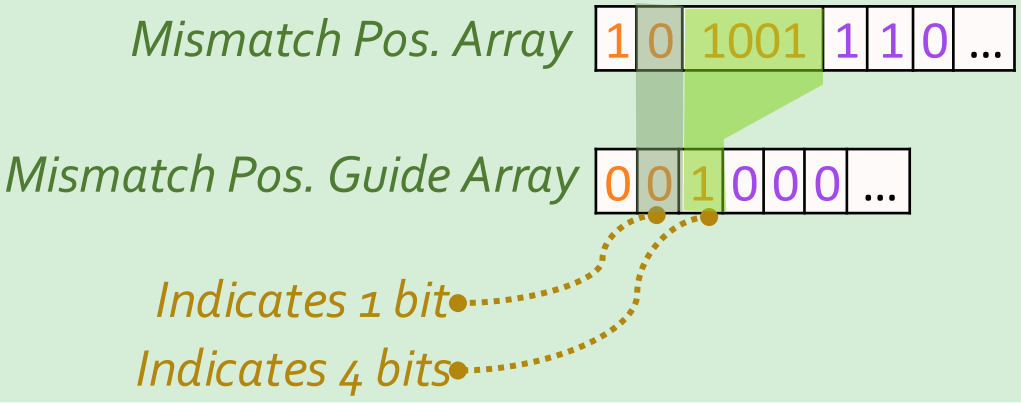
Mismatch Information

	Matching Pos.	Mismatch Pos.	Mismatch Base	Mismatch Type
Read 1	7	1, 0, 9	C, T, A	Subs, Subs, Ins
Read 2	10	1, 1, 0	A, T, T	Ins, Subs, Ins
...	...	...	...	...

① Sequentially Encode Mismatch Information in Lightweight Structures

1	0	9	1	1	0	
0001	0000	1001	0001	0001	0000	...

② Tune the Bit Counts Based on Read Set Properties

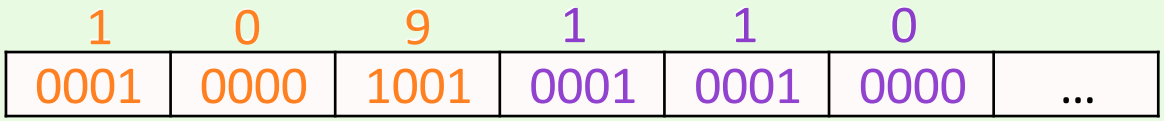


Compression Algorithm and Data Structures

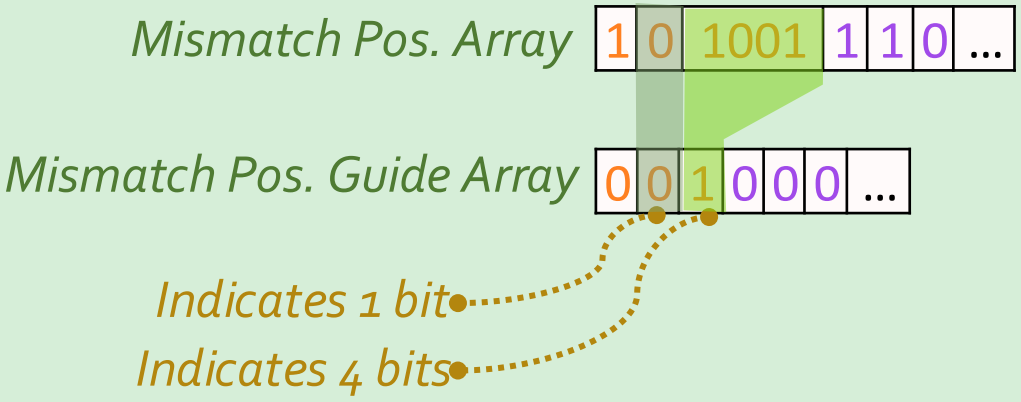
This is effective since the mismatch information in genomic datasets tends to follow specific properties

and by using a limited set of bit widths tailored to these properties, SAGe significantly reduces data size

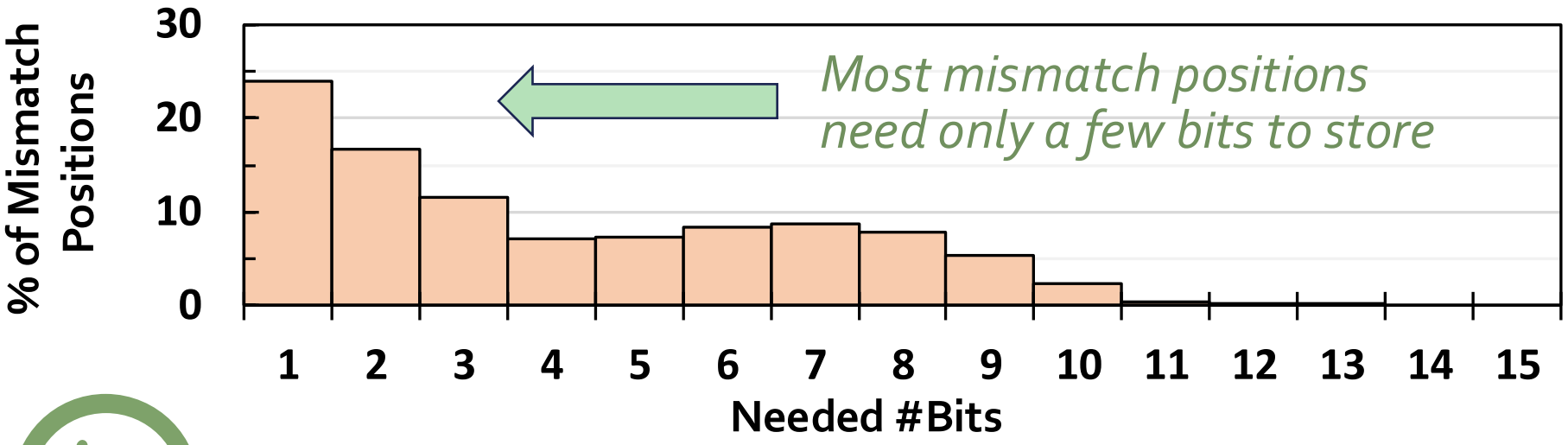
① **Sequentially Encode Mismatch Information in Lightweight Structures**



② **Tune the Bit Counts Based on Read Set Properties**



Example Properties of Mismatch Information



Based on the information encoded in histograms of ***each read set***,

SAGe systematically searches for
the **best bit-counts for arrays and guide arrays**

that **minimize the total encoded size** for that specific read set

Example Properties of Mismatch Information

Observations regarding properties of

other parts of the mismatch information

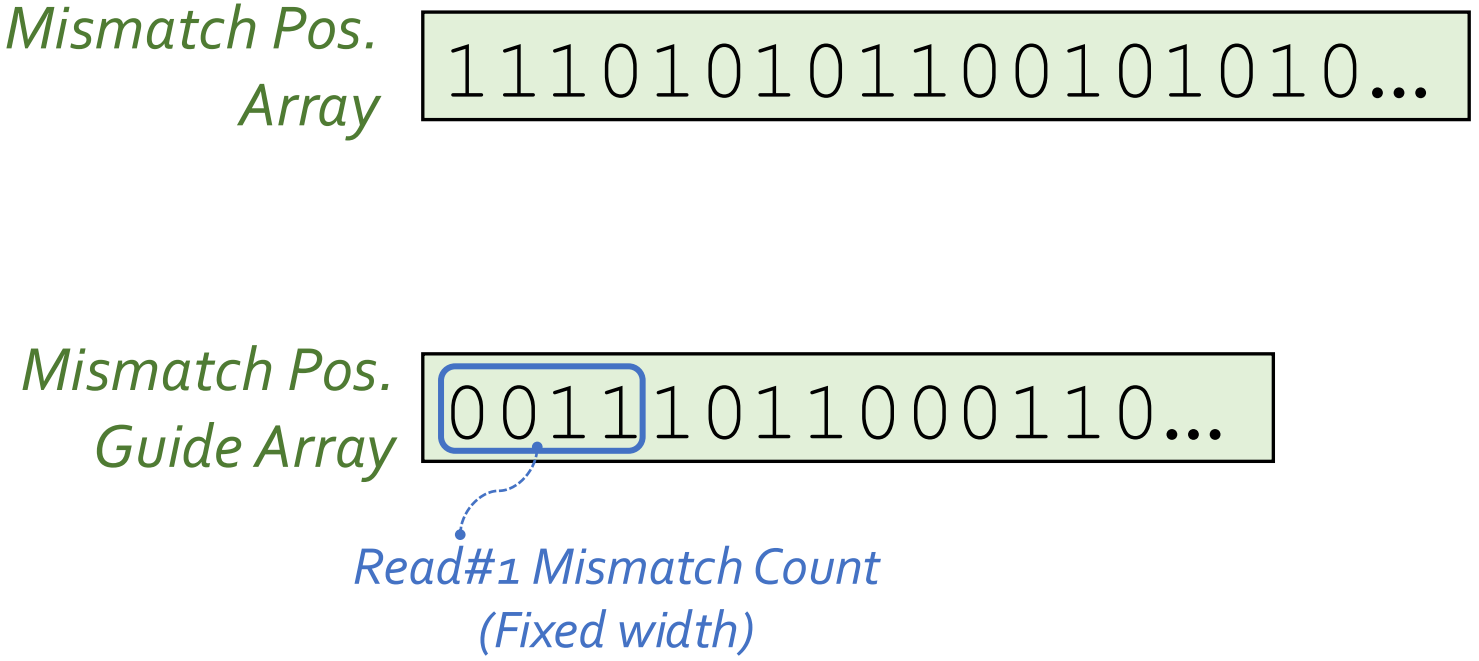
are in the paper

Based on the information encoded in histograms of
each read set,

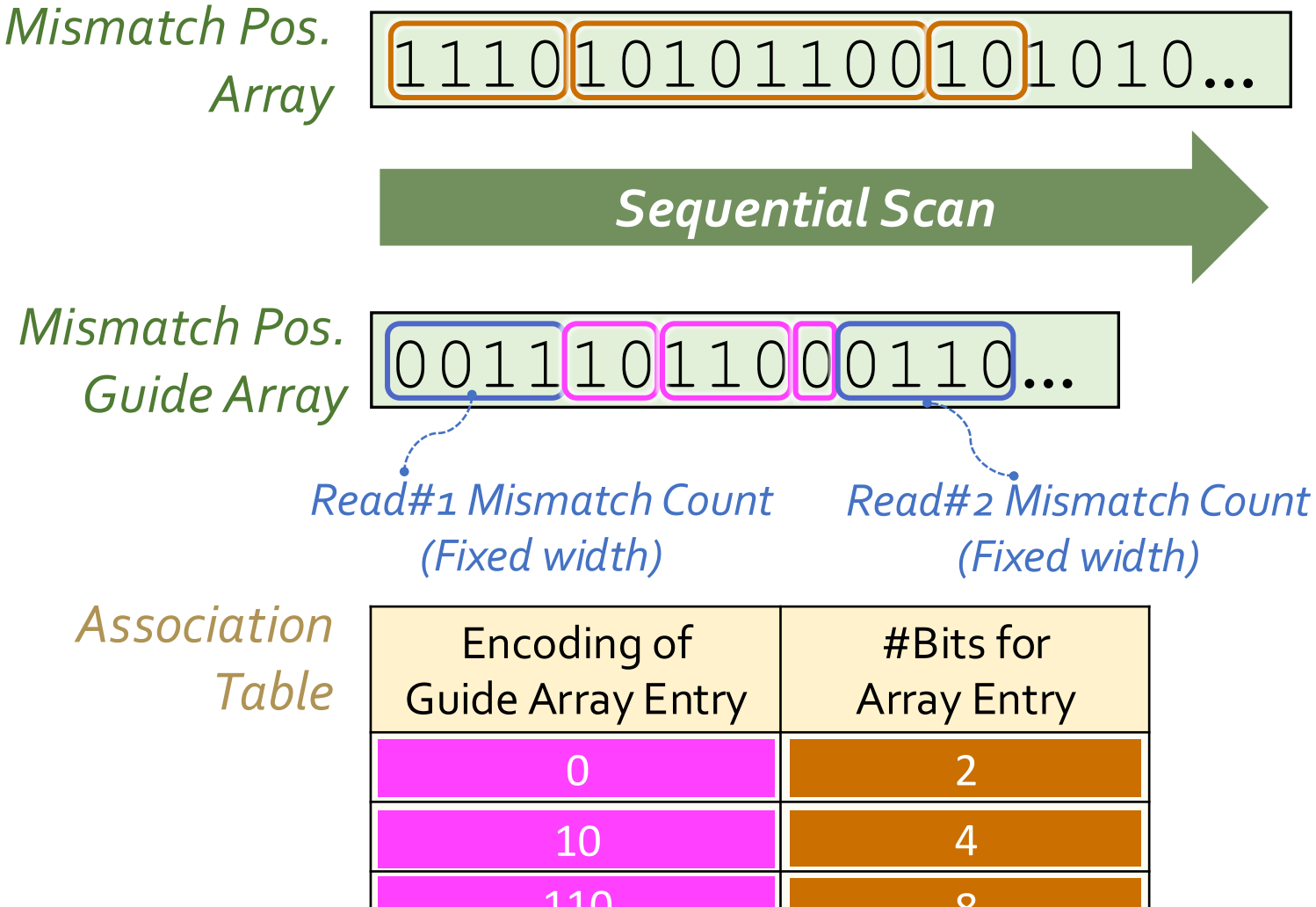
SAGe systematically searches for
the **best bit-counts for arrays and guide arrays**

that **minimize the total encoded size** for that specific read set

Example Walkthrough of Decompression



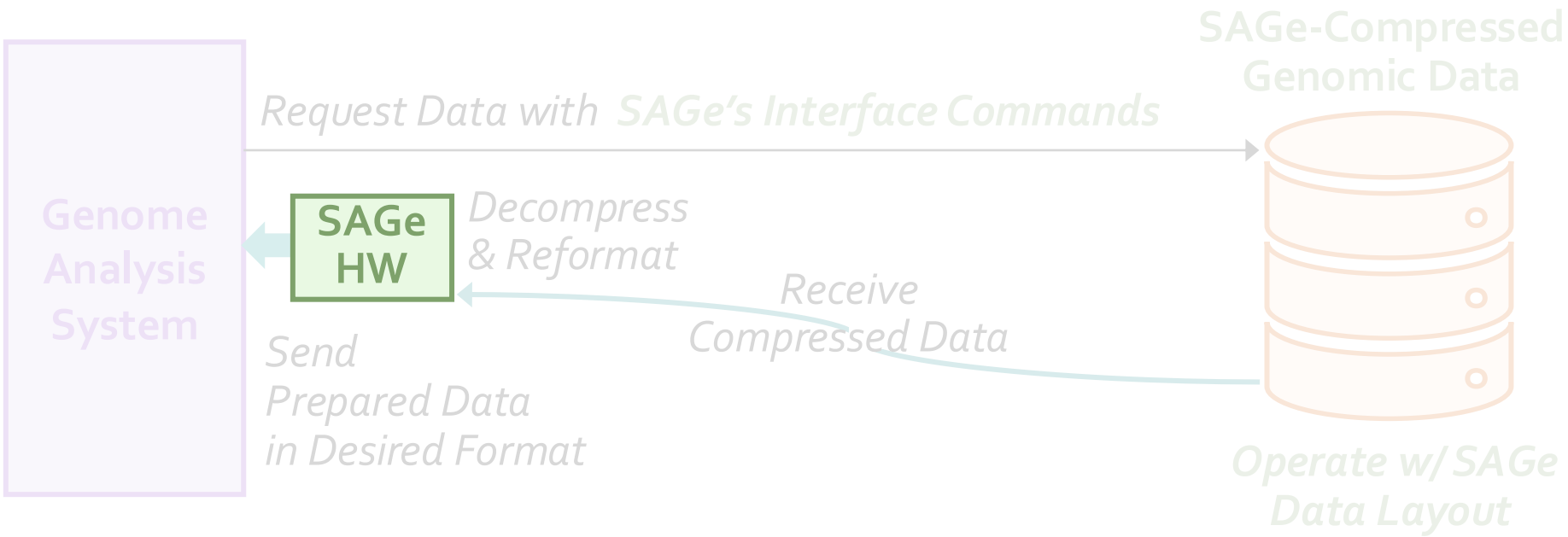
Example Walkthrough of Decompression



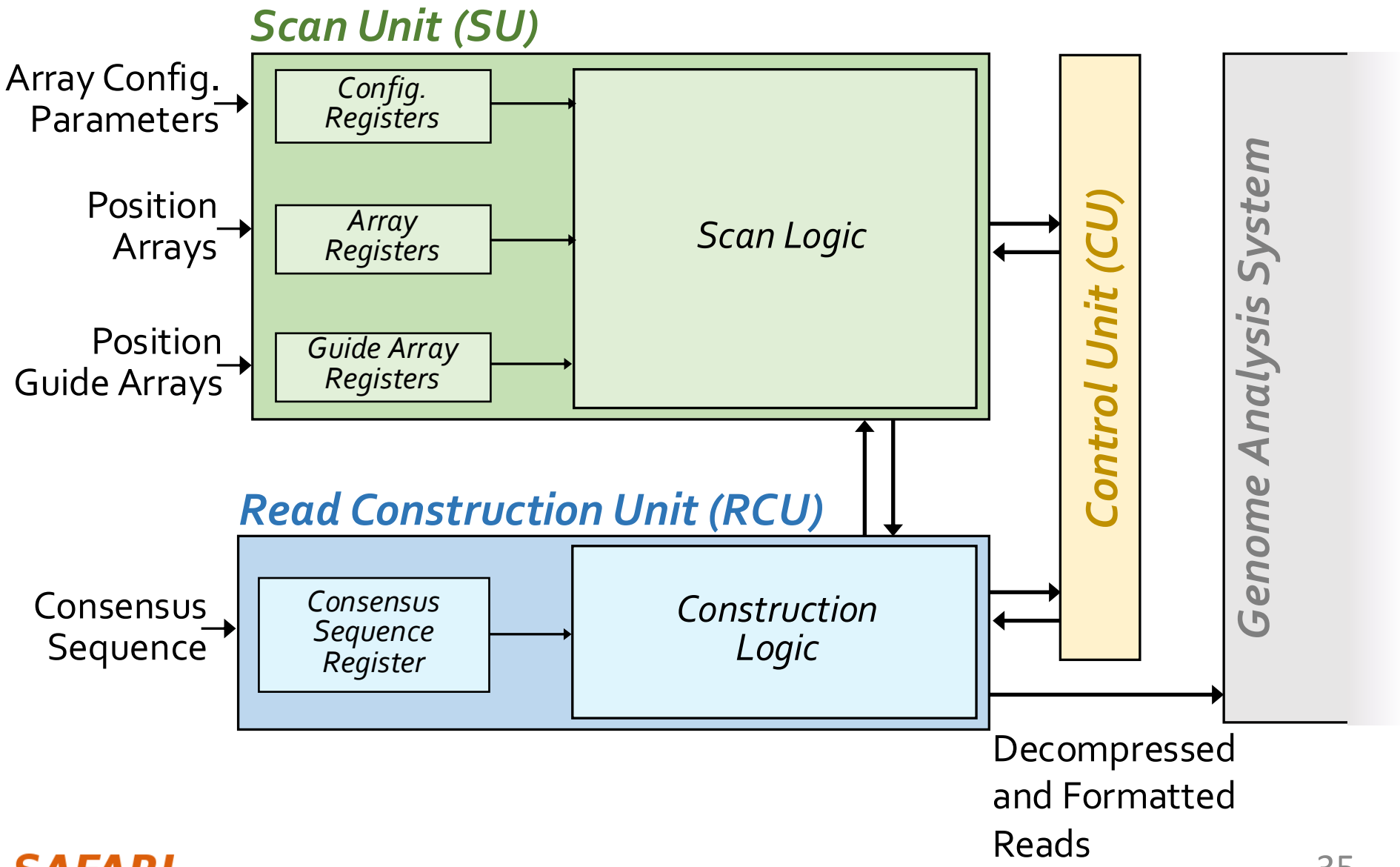
Association Table

Encoding of Guide Array Entry	#Bits for Array Entry
0	2
10	4
110	8

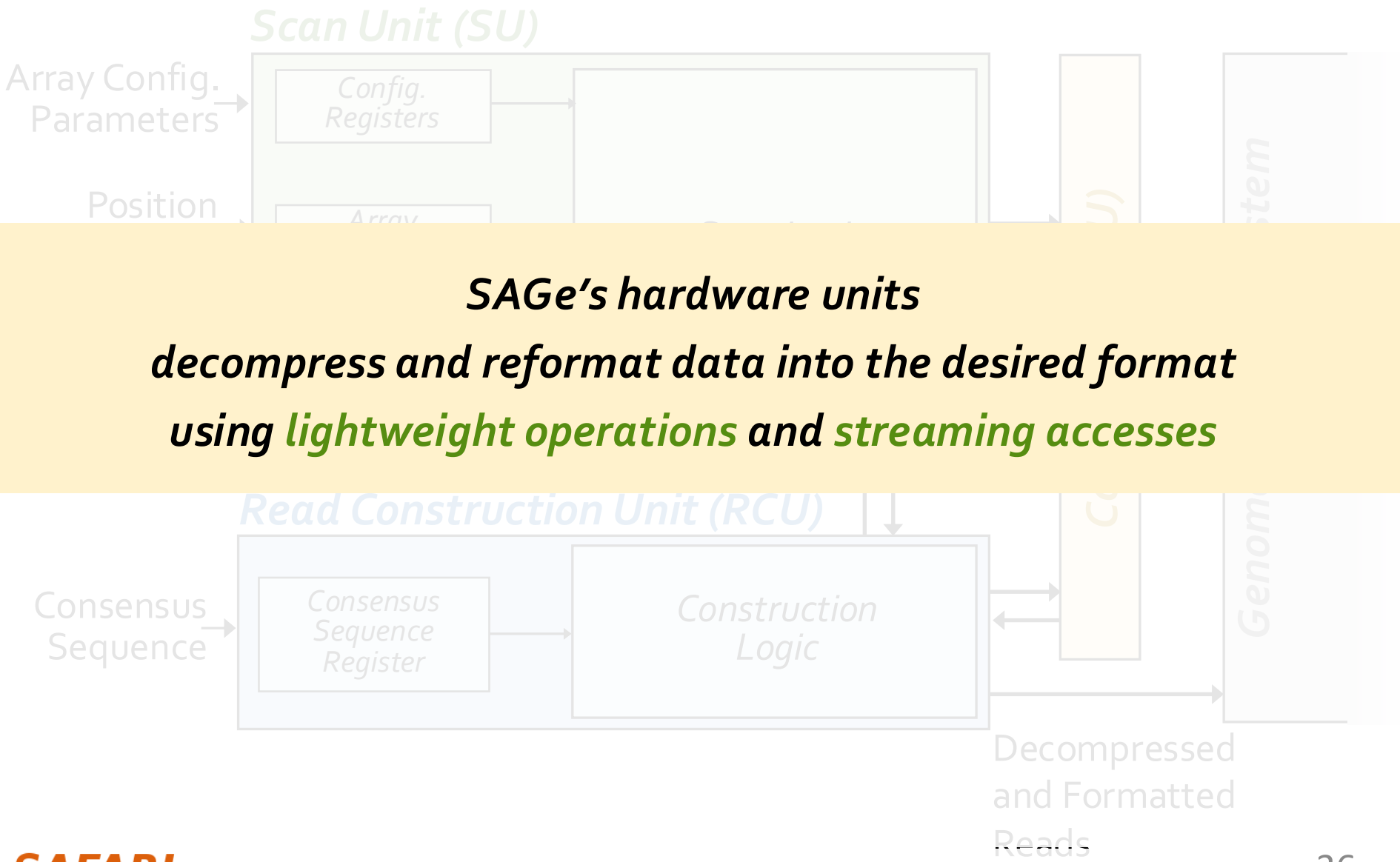
SAGe's Algorithm Architecture Co-Design



Hardware for Data Preparation



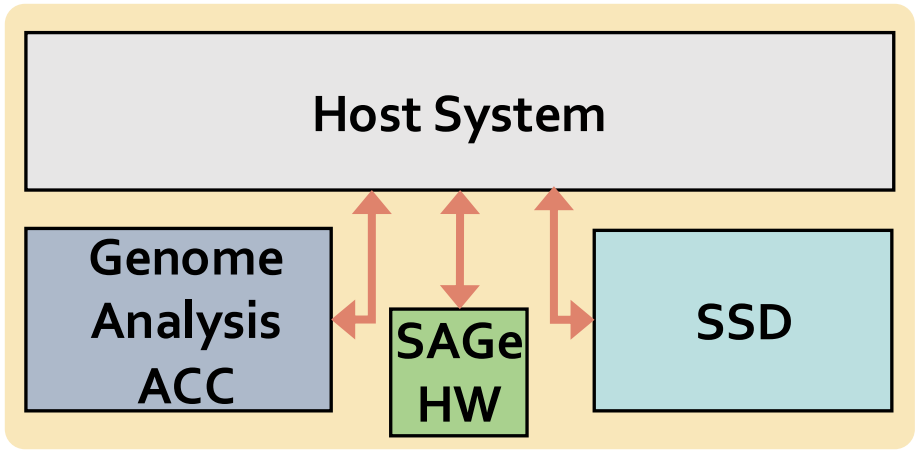
Hardware for Data Preparation

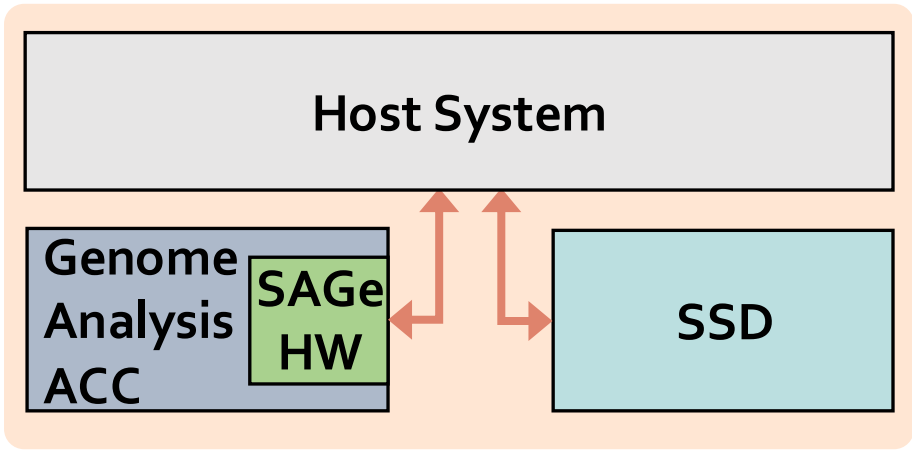


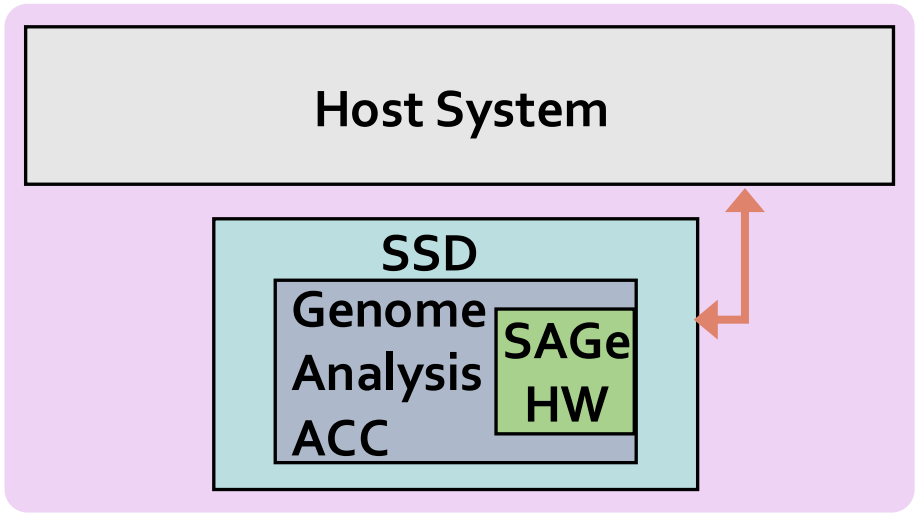
SAGe's hardware units decompress and reformat data into the desired format using lightweight operations and streaming accesses

Case Studies of SAGe's Hardware Integration

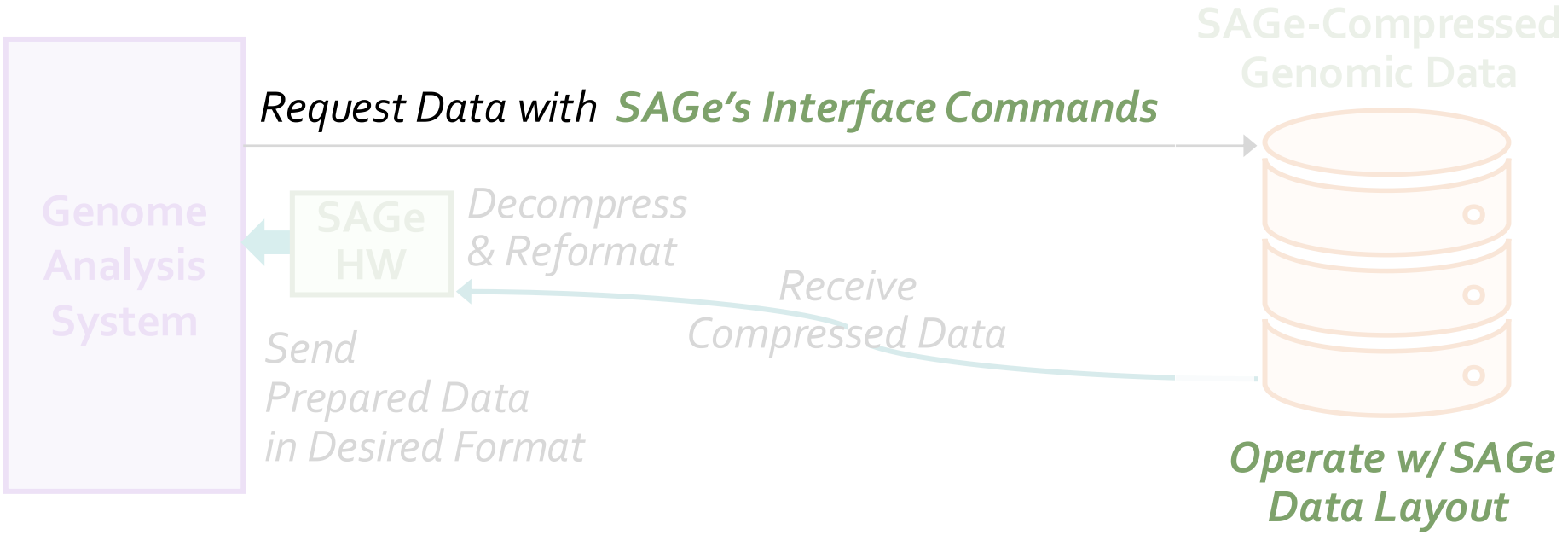
Due to its lightweight design, SAGe can seamlessly integrate with genome analysis systems







SAGe's Algorithm Architecture Co-Design



Details in the paper

More in the Paper

SAGe: A Lightweight Algorithm-Architecture Co-Design for Mitigating the Data Preparation Bottleneck in Large-Scale Genome Sequence Analysis

Nika Mansouri Ghiasi[1] Talu Güloglu[1] Harun Mustafa[1] Can Firtina[1,2]
Konstantina Koliogeorgi[1] Konstantinos Kanellopoulos[1] Haiyu Mao[3] Rakesh Nadig[1]
Mohammad Sadrosadati[1] Jisung Park[4] Onur Mutlu[1]

[1]ETH Zürich [2]University of Maryland [3]King's College London [4]POSTECH

https://arxiv.org/abs/2504.03732

Outline

Background

Motivation and Gaol

SAGe

Evaluation

Conclusion

Evaluation Methodology Overview (I)

Performance, Energy, and Power Analysis

Hardware Components

- Synthesized Verilog model for the hardware units
- MQSim *[Tavakkol+, FAST'18]* for SSD's internal operations
- Ramulator *[Kim+, CAL'15]* for SSD's internal DRAM

Software Components

Measure on a real system:

- AMD® EPYC® CPU with 128 physical cores
- 1 TB DRAM

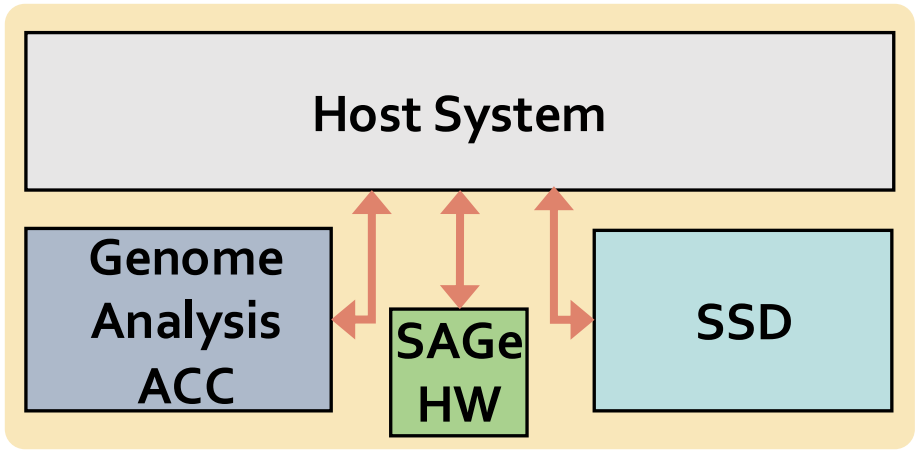
Datasets: Read sets (RS)

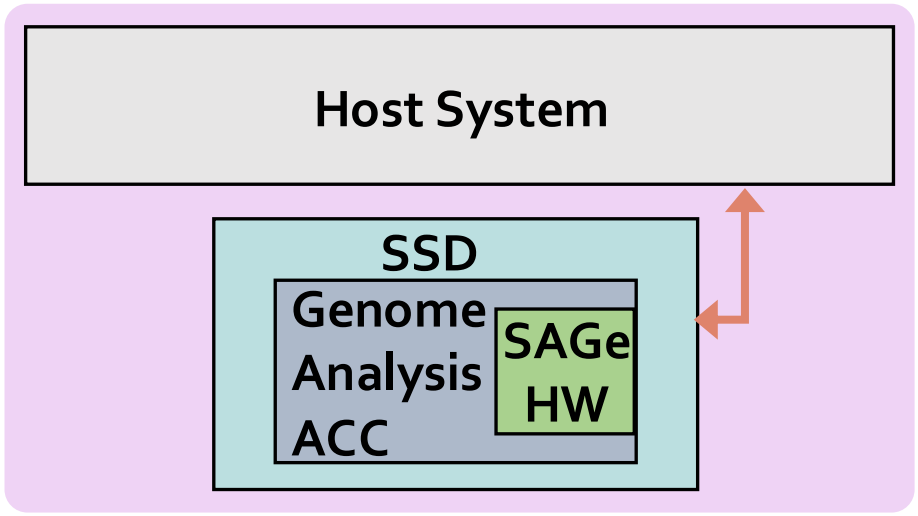
- From a variety of species (i.e., *H. sapiens*, *T. cacao*, *M. acuminata malaccensis*)
- Sequenced with different sequencing technologies (i.e., Illumina and Oxford Nanopore Technologies)

End-to-end performance of a genome analysis application, which includes both **data preparation** and **genome analysis**

Recall: Case Studies of SAGe's Hardware Integration

Due to its lightweight design, SAGe can seamlessly integrate with genome analysis systems

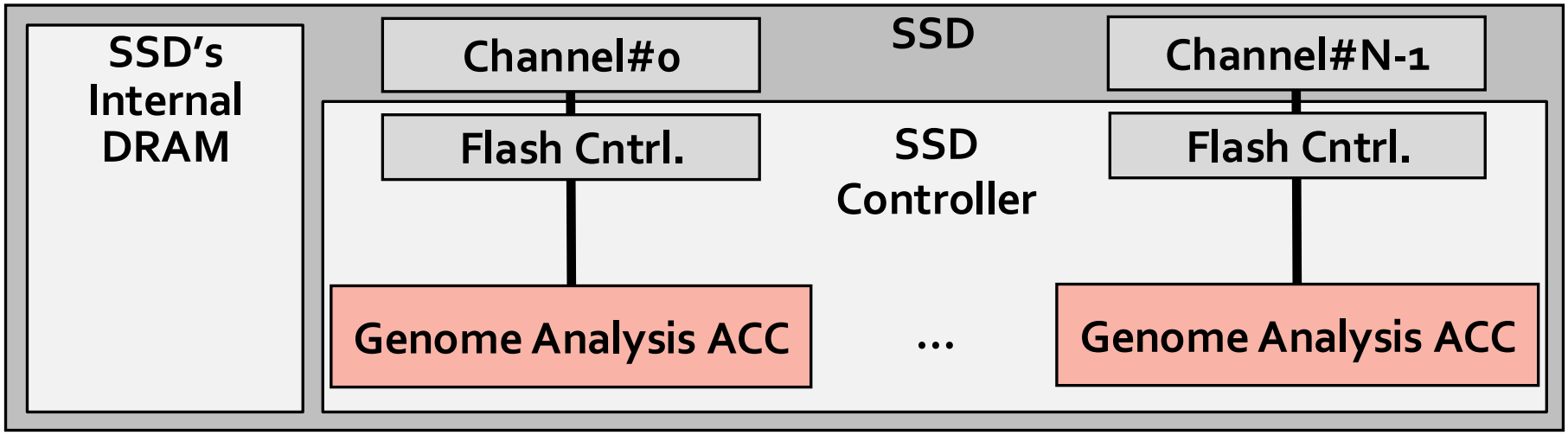




Evaluation Methodology Overview (II)

Genome Analysis

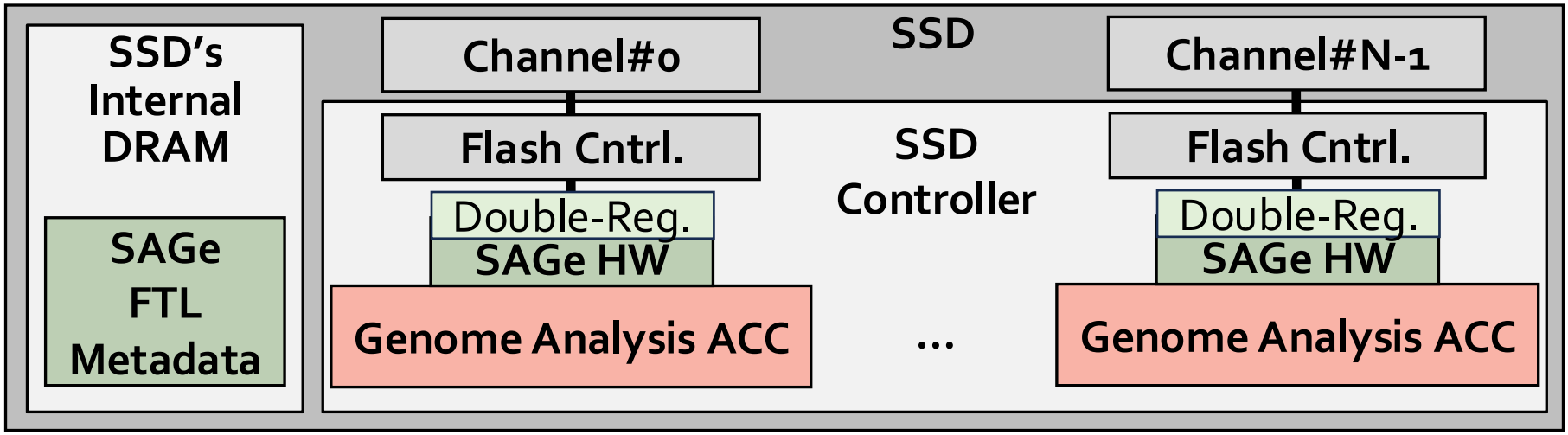
- A state-of-the-art hardware accelerator (GEM *[Chen+, TPDS'23]*), for read mapping
- A state-of-the-art NDP accelerator (GenStore *[Mansouri Ghiasi+, ASPLOS'22]*)
  - Filters reads that do not require expensive computation in the SSD ➔ Alleviates the burden of moving and analyzing a large amount of low-reuse data from the rest of the system
  - Implemented in the resource-constrained environment of the SSD



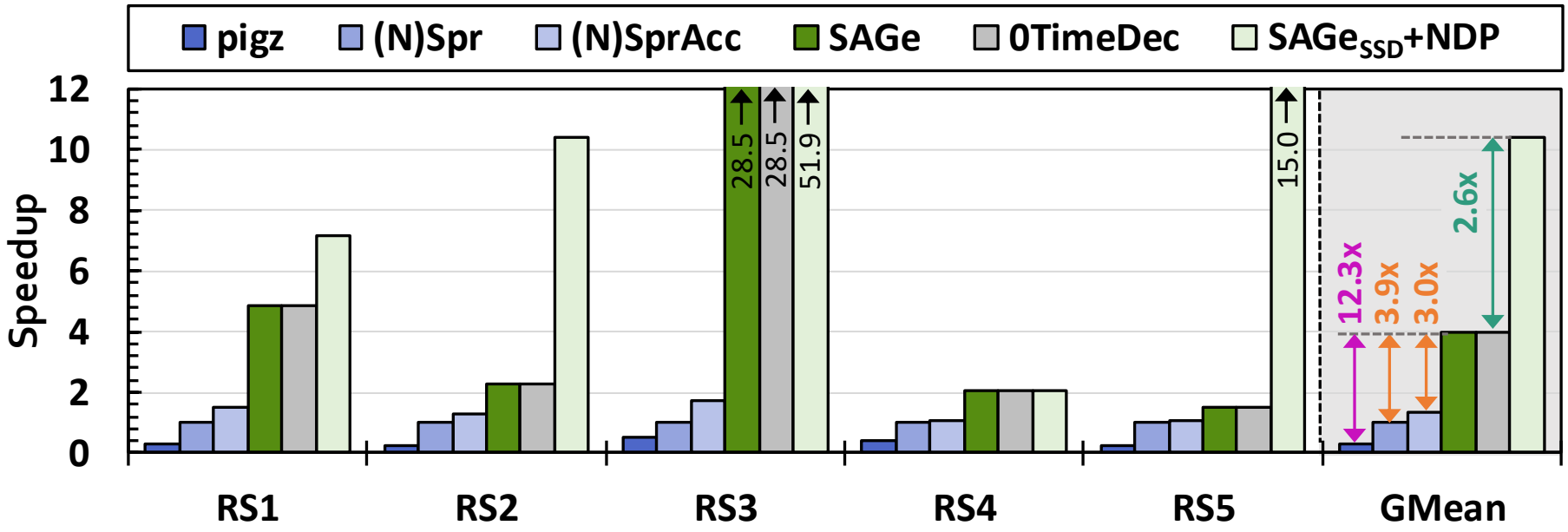
Evaluation Methodology Overview (II)

Data Preparation

- **pigz:** Commonly-used general-purpose software (*[Adler, JPL'15]*)
- **(N)Spr:** Genomics-specific software for short (Spring *[Chandak, Bioinformatics'18]*) and long reads (NanoSpring *[Meng+, Scientific Reports'23]*)
- **(N)SprAcc:** (N)Spring hardware-accelerated
- **0TimeDec:** an idealized decompressor (with *zero* decompression time), but inefficient for integration in resource-constrained environments
- **SAGe:** SAGe implemented as an standalone accelerator for data preparation
- **$SAGe_{SSD}$:** SAGe inside the SSD, for integration with the NDP analysis system



Speedup



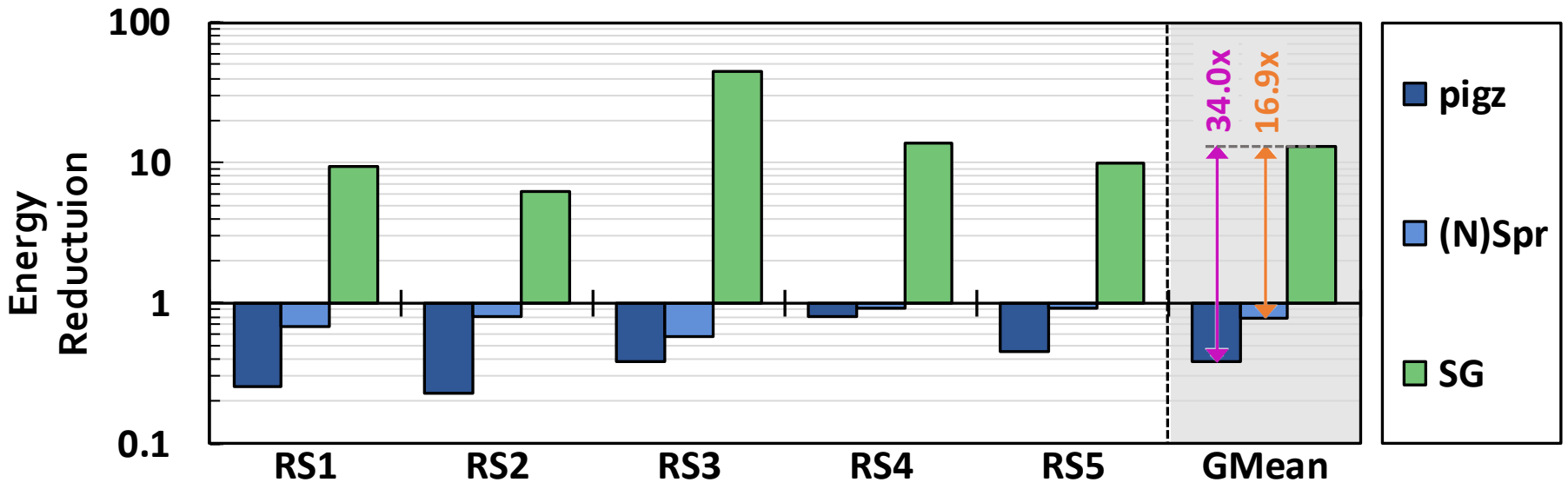
SAGe provides significant speedup over both general-purpose and genomics-specific baselines

SAGe achieves the same speedup as 0TimeDec

By efficiently integrating SAGe$_{SSD}$ with NDP SAGe$_{SSD}$+NDP leads to large speedups

Reduction in Energy Consumption

Normalized to (N)SprAcc (higher is better)



SAGe provides significant energy reduction over both general-purpose and genomics-specific baselines

Compression Ratio

Read Set	Uncompressed Size (MB)	Compression Ratio		
		pigz	(N)Spr	SAGe
RS1	5000	3.4	24.8	22.8
RS2	79000	12.5	40.2	36.8
RS3	4000	3.4	7.2	7.1
RS4	12000	3.9	4.8	4.5
RS5	88400	3.5	7.6	7.8

2.9× better average compression ratio than general-purpose baseline

comparable compression ratio to genomics-specific baseline
(with a modest 4.6% average reduction)

Area and Power

- Based on **Synthesis** of **SAGe's hardware units** using the Synopsys Design Compiler @ 22nm technology node

Logic unit	# of instances	Area [mm²]	Power [mW]
Comparator	1 per channel	0.000045	0.014
Read Construction Unit	1 per channel	0.000017	0.023
Double Registers	1 per channel	0.00020	0.035
Control Unit	1 per channel	0.000029	0.025
Total for an 8-channel SSD	-	0.002	0.77

SAGe's hardware units consume small area and power

More in the Paper

- Observations on different parts of **mismatch information**
  - Matching positions
  - Mismatch bases and types
  - Mismatch counts
- SAGe's performance in **software**
- Sensitivity analysis with **varying the number of SSD**
- Impact of **each of SAGe's optimizations** on compression ratio
- SAGe's performance for **quality scores**
- SAGe's **resource requirements** compared to other (de)compression accelerators
- SAGe's compression time

More in the Paper

SAGe: A Lightweight Algorithm-Architecture Co-Design for Mitigating the Data Preparation Bottleneck in Large-Scale Genome Sequence Analysis

Nika Mansouri Ghiasi[1] Talu Güloglu[1] Harun Mustafa[1] Can Firtina[1,2]
Konstantina Koliogeorgi[1] Konstantinos Kanellopoulos[1] Haiyu Mao[3] Rakesh Nadig[1]
Mohammad Sadrosadati[1] Jisung Park[4] Onur Mutlu[1]

[1]ETH Zürich [2]University of Maryland [3]King's College London [4]POSTECH

https://arxiv.org/abs/2504.03732

Outline

Background

Motivation and Gaol

SAGe

Evaluation

Conclusion

Conclusion

Data preparation bottleneck
greatly limits end-to-end performance of genome analysis

SAGe

An algorithm-architecture co-design for highly-compressed storage and high-performance access of large-scale genomic sequence data

Improves performance

Improves the end-to-end performance of two state-of-the-art genome analysis accelerators by 3.0×–32.1×

High compression ratio

2.9× better than general-purpose baseline and comparable to genomics-specific baseline (with a modest 4.6% average reduction)

Reduces energy consumption

Improves the end-to-end energy efficiency of two state-of-the-art genome analysis accelerators by 13.0×–34.0×

Low overhead

Area: 0.002 mm^2

Power: 0.77 mW

A Lightweight Algorithm-Architecture Co-Design for Mitigating the Data Preparation Bottleneck in Large-Scale Genome Sequence Analysis

Nika Mansouri Ghiasi

Talu Güloglu Harun Mustafa Can Firtina Konstantina Koliogeorgi

Konstantinos Kanellopoulos Haiyu Mao Rakesh Nadig

Mohammad Sadrosadati Jisung Park Onur Mutlu

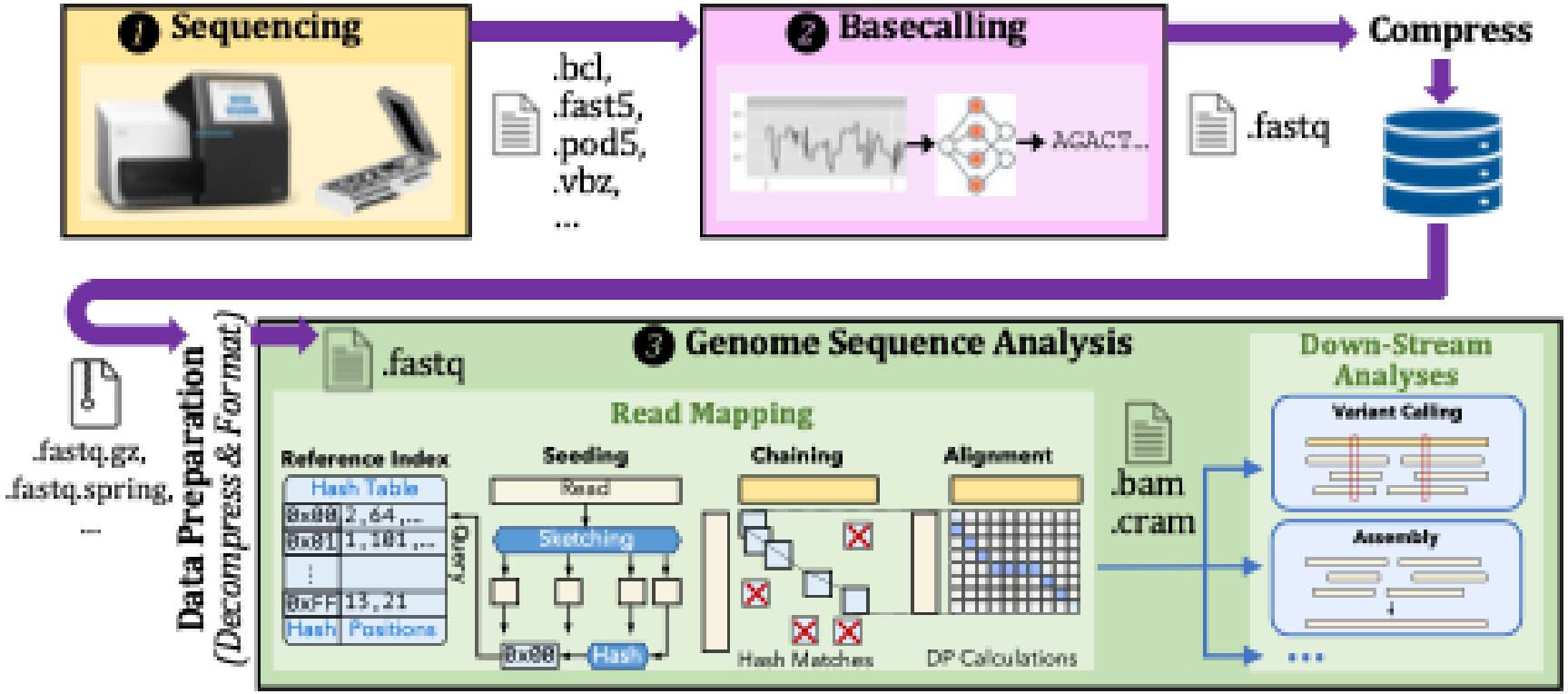
SAFARI

ETH zürich UNIVERSITY OF MARYLAND KING'S College LONDON POSTECH

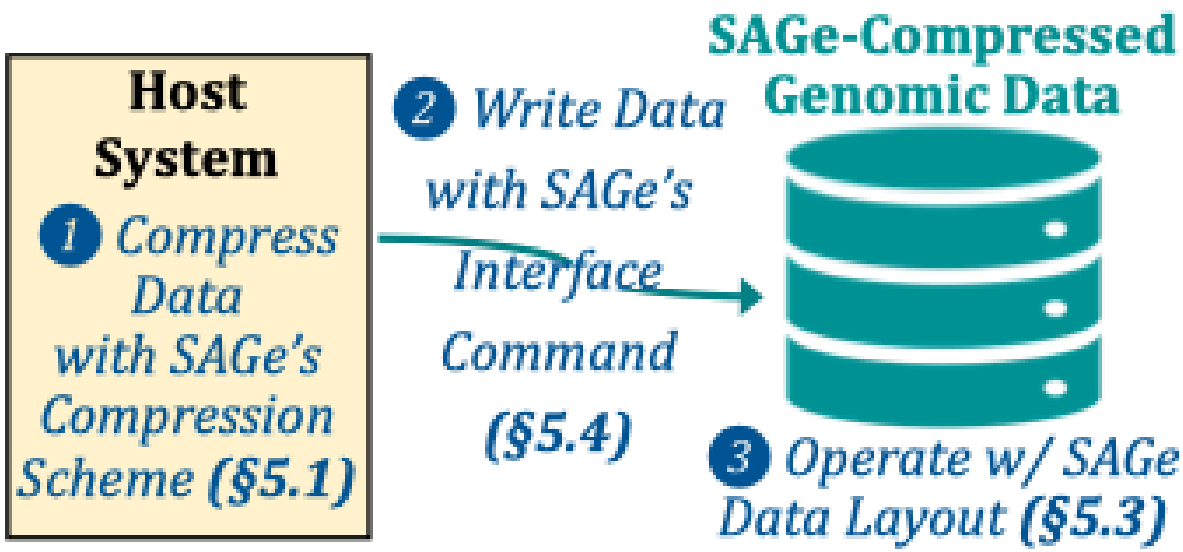
Backup Slides

- Genomic Workflow
- SAGe Compression
- Properties of Mismatch Information
- Tuning Arrays and Guide Arrays
- Hardware Details
- Data Preparation Performance
- End-to-End Performance
- Performance with Multiple SSDs
- Compression Ratio Details
- Effect of Different Optimizations
- Resource Requirements
- Compression Time

Genomic Workflow

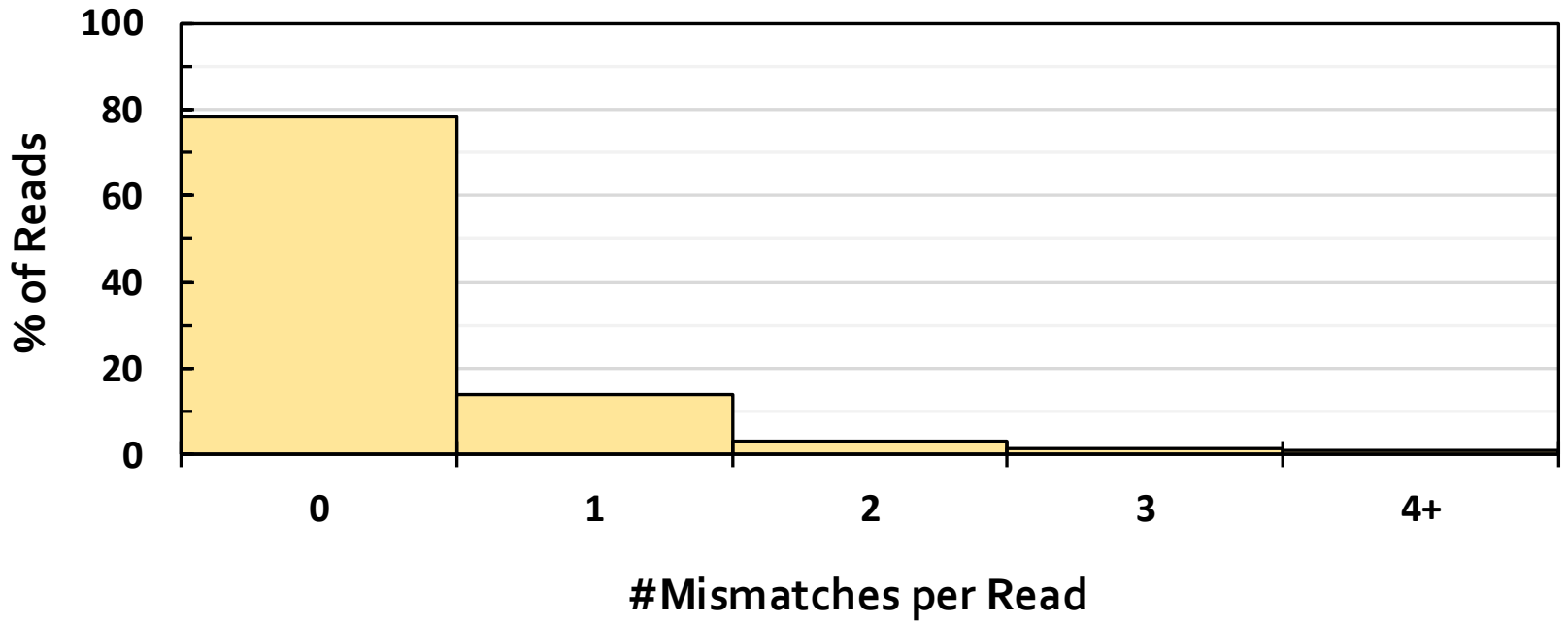


SAGe Compression High-Level Overview



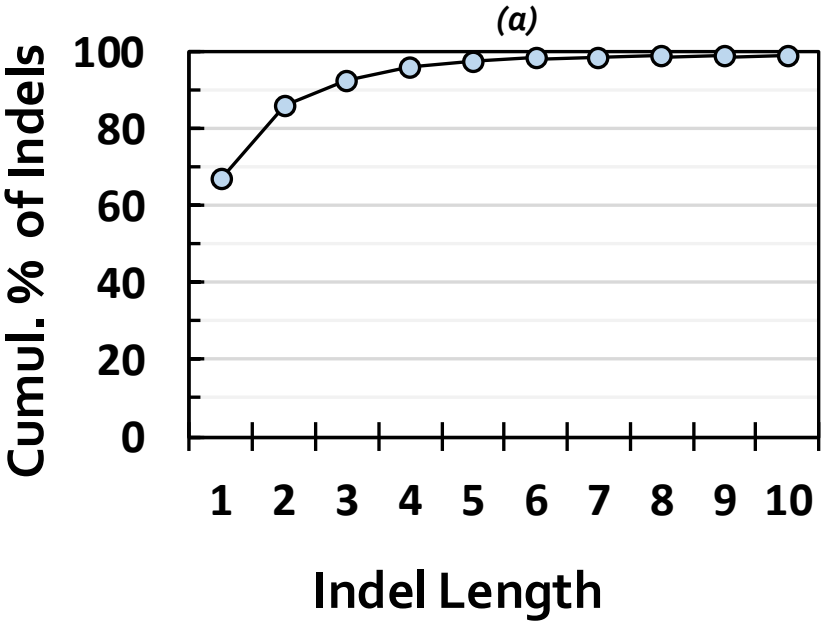
Properties of Mismatch Information (I)

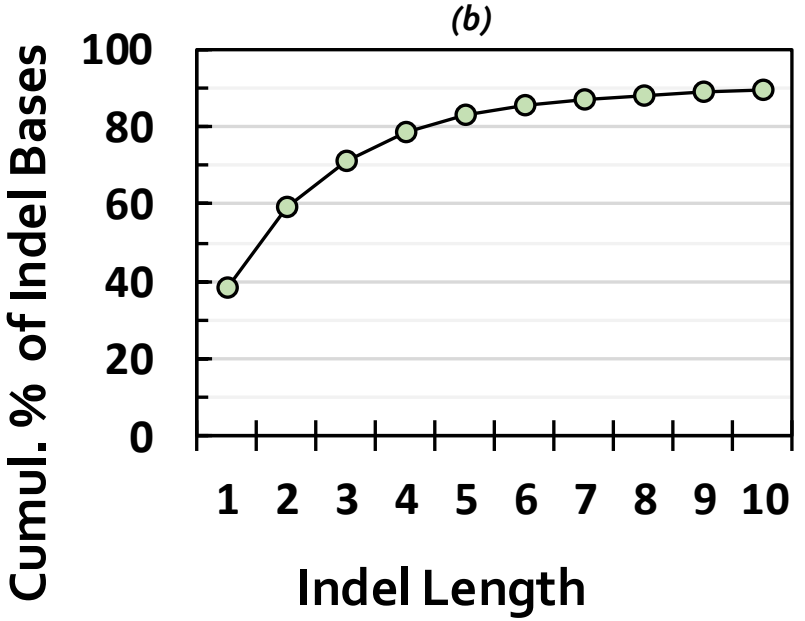
Distribution of mismatch counts per short read



Properties of Mismatch Information (II)

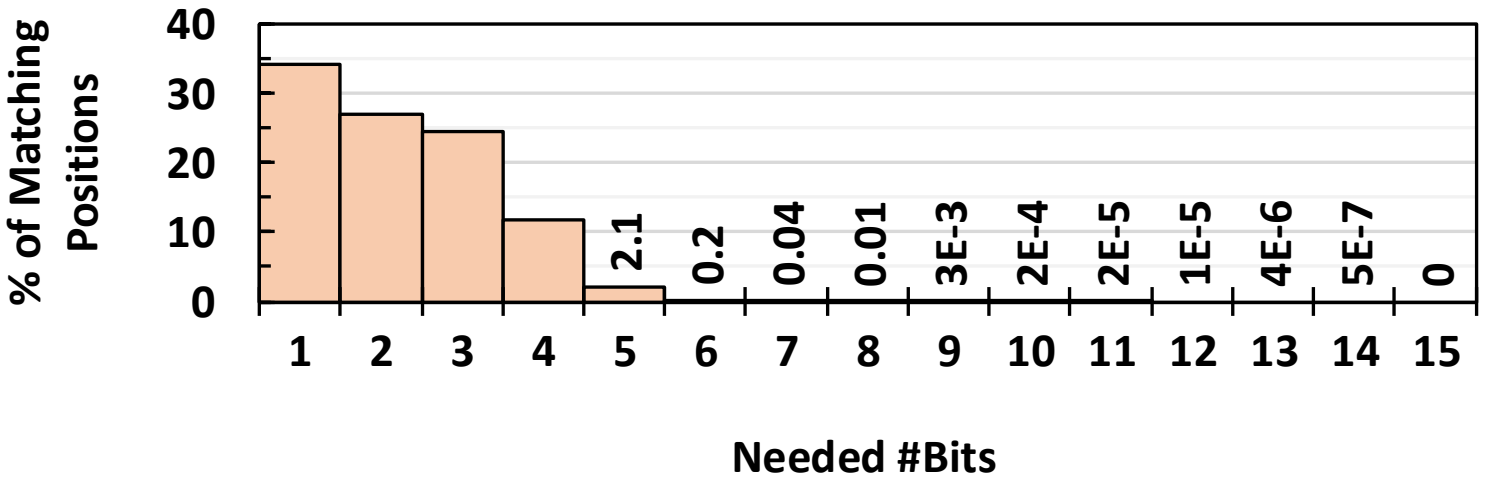
Cumulative distribution of
(a) indel block lengths and
(b) #bases in indel blocks of different lengths





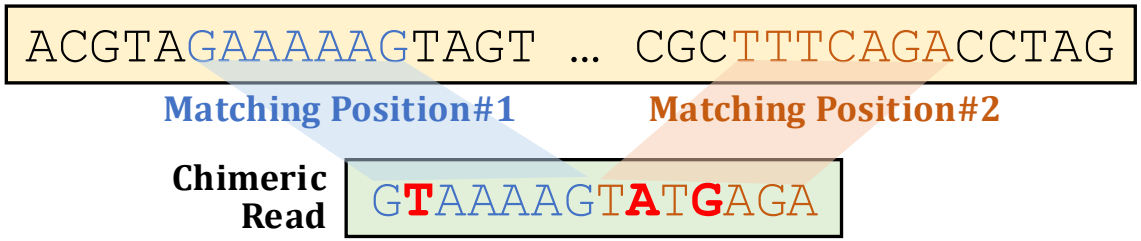
Properties of Mismatch Information (III)

Distribution of #bits needed to store the delta-encoded matching position.



Example of a chimeric read

Chimeric reads: reads with sequences joined from different regions of the genome due to sequencing/library preparation errors, or structural variations



Tuning Arrays and Guide Arrays

Algorithm 1 Tuning Bit Counts

Input: Histogram of mismatch position bit counts H, where $|H| \leq 32$, and a tuning convergence threshold ε.

Output: List of bit counts W for encoding mismatch positions.

$\ell_{\text{min}} \leftarrow \infty$ ▷ *Initialize min. encoding size*

$x_0 \leftarrow 0$

for all $d \in \{1, \ldots, 8\}$ **do** ▷ *Find optimal $|W|$.*

  $\ell_{\text{last}} \leftarrow \ell_{\text{min}}$ ▷ *Best result from previous d values*

  for all $(x_1, \ldots, x_d) \in |H|^d$ s.t. $x_0 < x_1 < \cdots < x_d$ **do**

    $\ell \leftarrow$ total length of encoded mismatch positions and guide arrays s.t. positions with bit counts $\in (x_i, x_{i+1}]$ are encoded with x_{i+1} bits.

    if $\ell < \ell_{\text{min}}$ **then**

      $\ell_{\text{min}} = \ell$

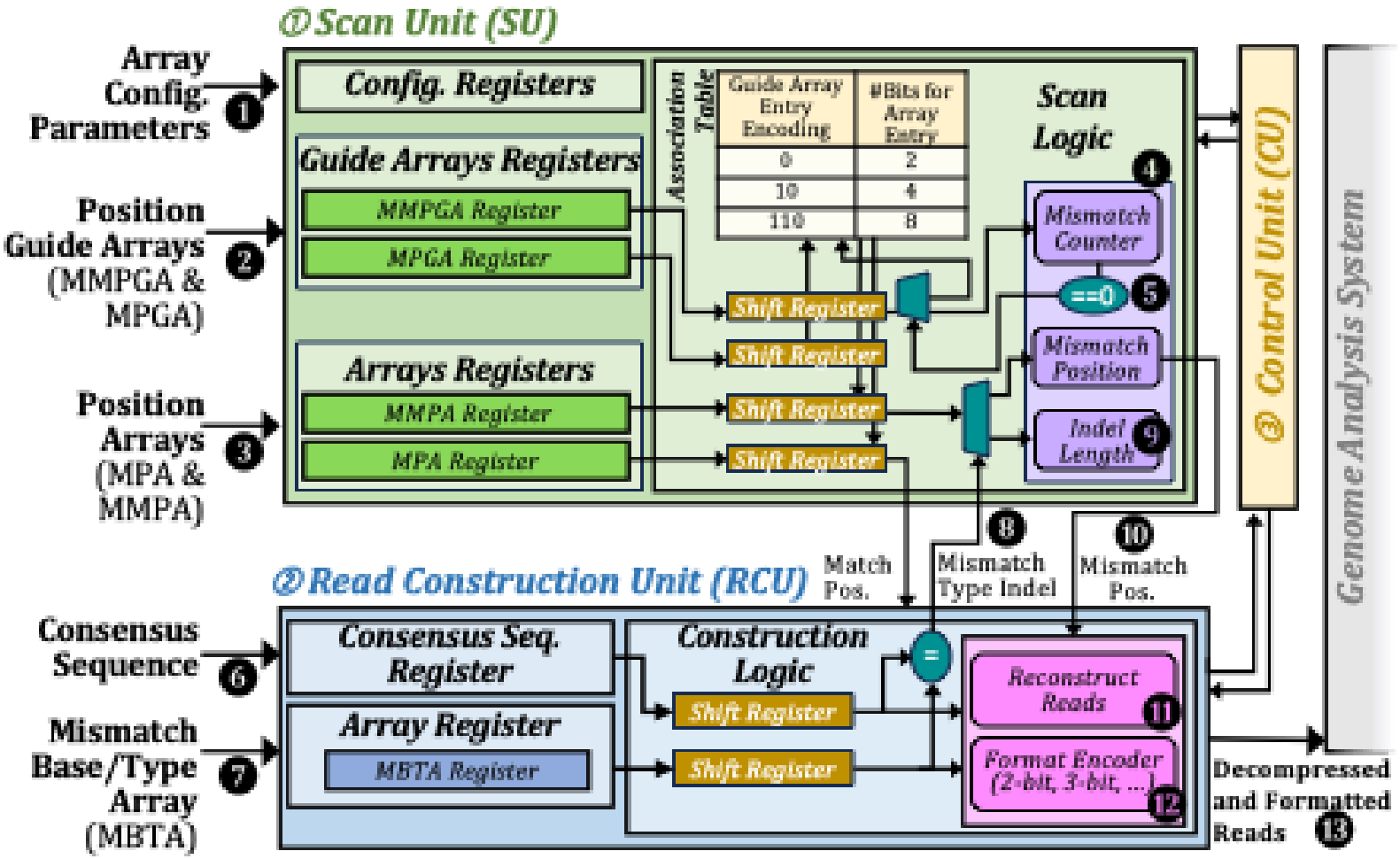
      $W \leftarrow (x_1, \ldots, x_d)$

  if $(\ell_{\text{last}} - \ell_{\text{min}})/\ell_{\text{min}} < \varepsilon$ **then**

    break ▷ *Exit loop when ℓ_{min} converges, typically at $d < 8$*

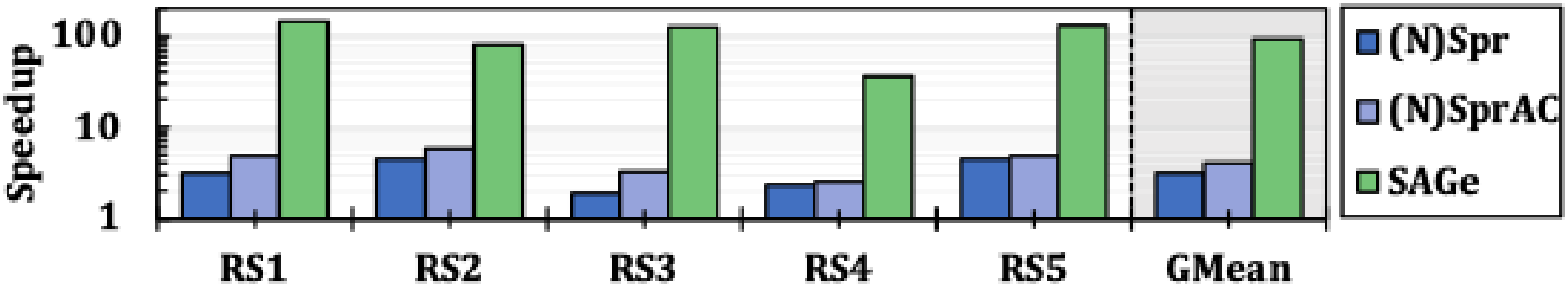
return W

Hardware for Data Preparation



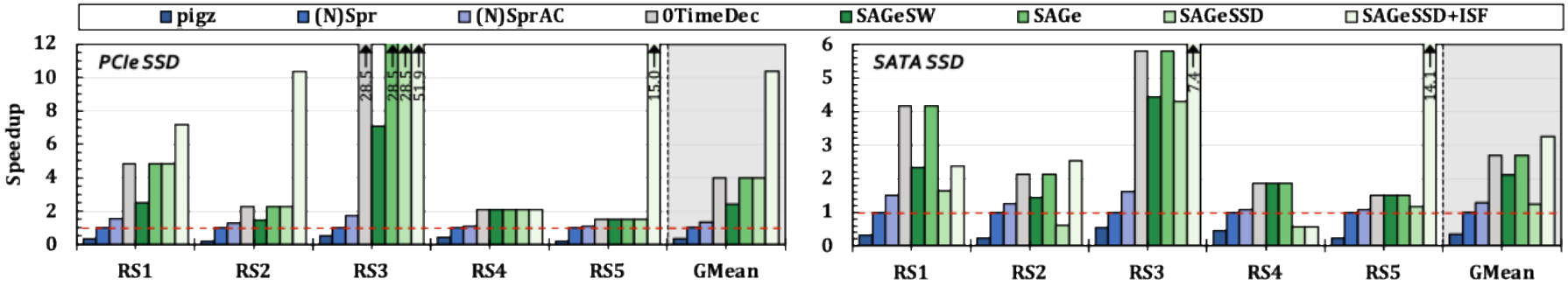
Data Preparation Performance

Only data preparation throughput, normalized to pigz

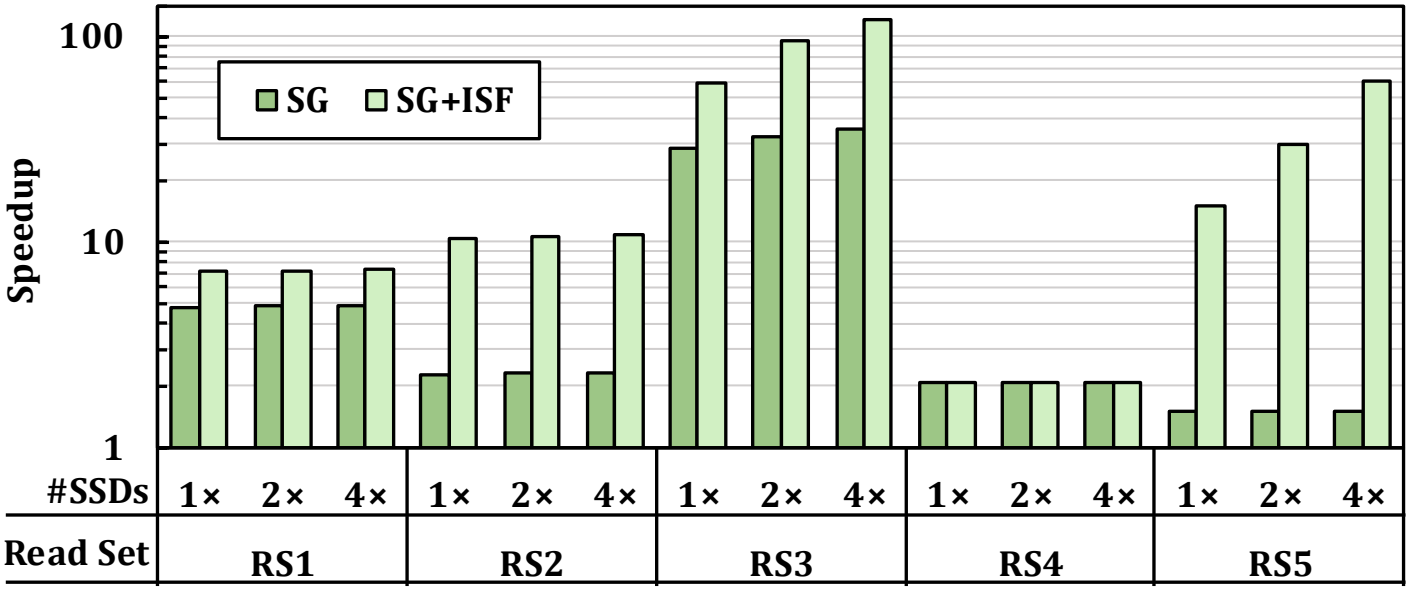


End-to-End Performance

Speedup, with different SSDs



Performance with Multiple SSDs



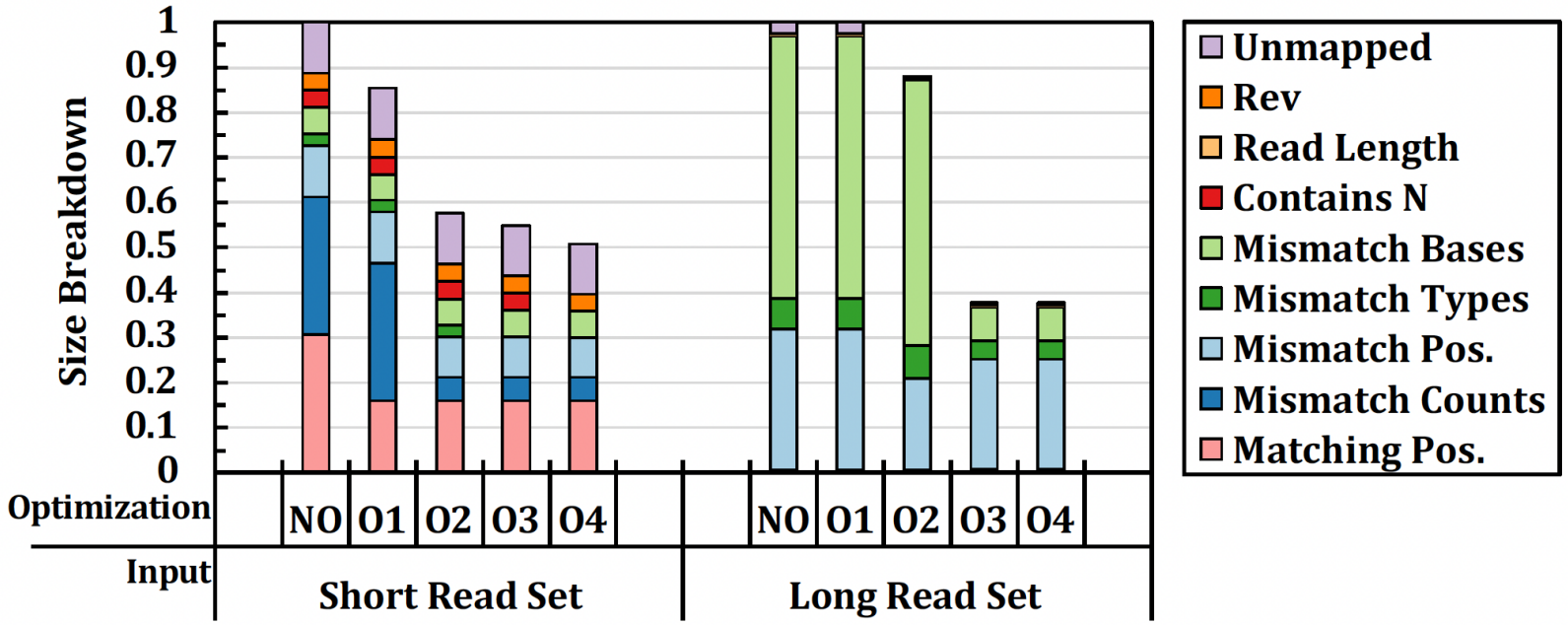
Compression Ratio

Table 2: Compression ratios for different read sets.

Label	Read Set (Short, Long)	Uncomp. Size (MB) DNA+Qual	Compression Ratio					
			PigZ [167]		(Nano)Spring [48]		SAGe	
			DNA	Qual.	DNA	Qual.	DNA	Qual.
RS1	SRR870667_2 [342] (S)	10 000	3.39	2.23	24.8	2.80	22.8	2.80
RS2	ERR194146_1 [343] (S)	158 000	12.5	2.49	40.2	3.4	36.8	3.4
RS3	SRR2052419_1 [344] (S)	8 000	3.41	3.45	7.2	5.07	7.1	5.07
RS4	PAO89685_sampled [345] (L)	24 000	3.93	1.79	4.8	2.19	4.5	2.19
RS5	ERR5455028 [346] (L)	176 800	3.5	1.57	7.6	1.82	7.8	1.82

Effect of Different SAGe Optimizations

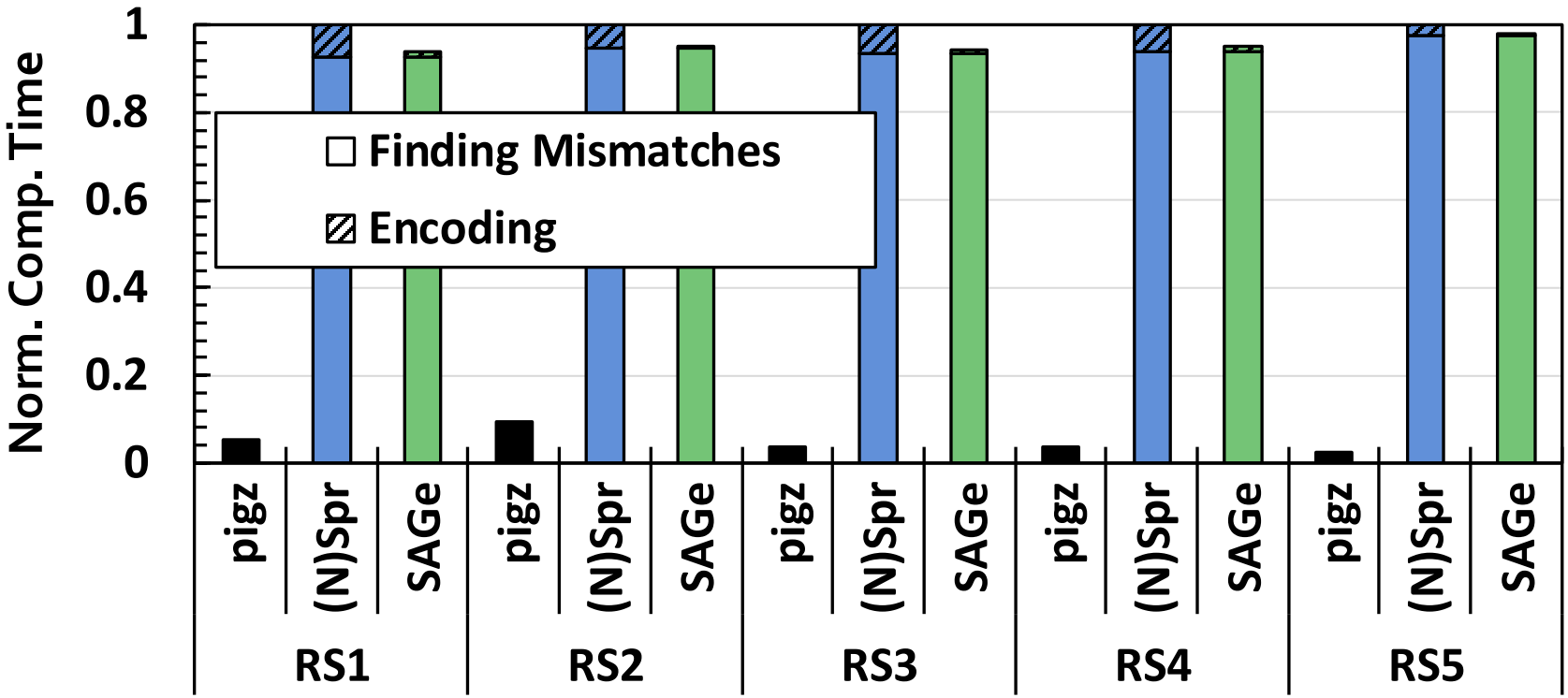
NO: no optimization on the raw mismatch information
O1: matching position optimization added to NO
O2: mismatch positions and count optimizations added to O1
O3: mismatching base and type optimizations added to O2
O4: corner case optimizations added to O3



Resource Requirements

Tool	Genomics-specific	Avg. Comp. Ratio	End-to-end?	Hardware Requirements	Memory Footprint	Decomp. Throughput (GB/s)
nvCOMP (DEFLATE) [347]	×	5.3	✓	GPU (NVIDIA A100 [348])	1.5 GB	50
Xilinx/AMD GZIP Engine [349]	×	5.3	✓	FPGA (AMD Alveo U50 [350])	80 KB	0.7
xz [274]	×	6.7	✓	CPU (AMD® EPYC® 7742 CPU [324] with 128 physical cores)	13 GB	0.6
HW-accelerated ZStandard [351]	×	6.7	✓	ASIC ($1.89\ mm^2$ in a 14nm technology node)	2–64 KB	3.9
GPUFastqLZ [53]	✓	5.8	✓	GPU (A GPU server with four NVIDIA Tesla V100 GPUs [352])	N/A[14]	7.8
repaq [54]	✓	17.1	×	FPGA (AMD ALVEO U200 [353])	16 GB	N/A
Spring [43] & NanoSpring [48]	✓	16.9	✓	CPU (AMD® EPYC® 7742 CPU [324] with 128 physical cores)	26 GB	0.7
SAGe	✓	15.8	✓	ASIC ($0.002\ mm^2$ in a 22nm technology node)	128 B	75.4

Compression Time

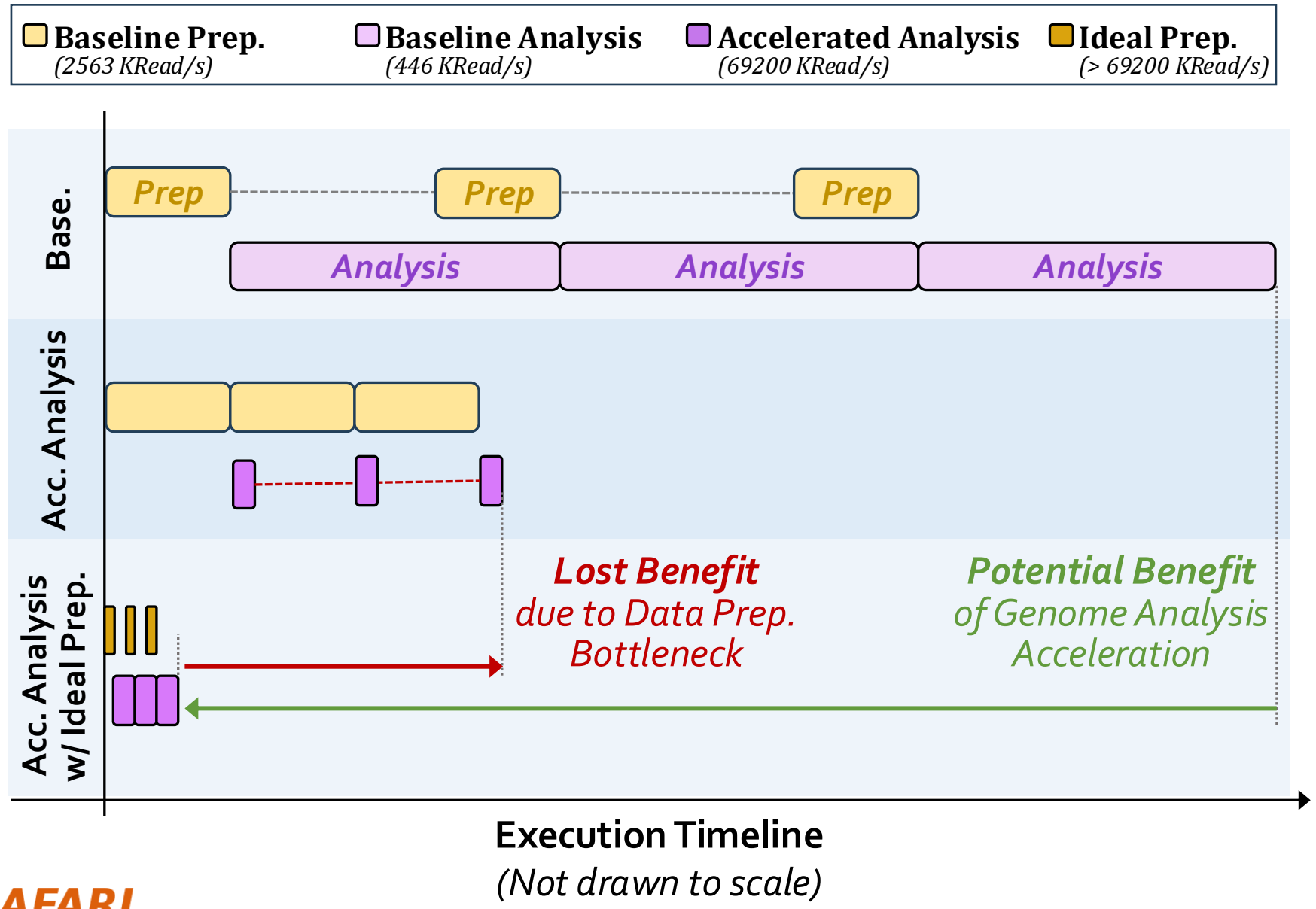


SAGe

A Lightweight Algorithm-Architecture Co-Design for Mitigating the Data Preparation Bottleneck in Large-Scale Genome Sequence Analysis

When: *Tuesday (Feb 3) @ 17:35 AEDT*
Where: *ICC Sydney, Coogee Room*

Effect of the Data Preparation Bottleneck



SAGe

*An algorithm-architecture co-design for highly-compressed **s**torage and high-performance **a**ccess of large-scale **ge**nomic sequence data*

- **Key insight**: information encoded in genomic data follows specific properties
- We leverage these properties to co-design
  - A lossless (de)compression algorithm
  - Hardware for decompression with lightweight operations
  - Storage data layout & Interface commands to access data

Higher performance *(by 3.0×–32.1×)*

Higher energy efficiency *(by 13.0×–34.0x)*

High compression ratio

Low overhead

SAFARI