

Storage-Centric System Designs for Enabling Fast, Efficient, and Low-Cost Genomic and Metagenomic Analyses

Nika Mansouri Ghiasi

Onur Mutlu

Arch4Health @ ICS 2026
June 2026

SAFARI

ETH zürich

Brief Self-Introduction

- **Nika Mansouri Ghiasi**

- Researcher and Lecturer in the SAFARI Research Group @ ETH Zurich
 - Defended my PhD thesis, advised by Onur Mutlu, on 31 March 2026
- Visiting researcher at Robust Systems Group @ Stanford

- **Research interests:**

- Computer architecture
- Computational biology
- Storage systems
- Emerging technologies, esp. dense 3D integration
- ...

*Designing **high-performance, energy-efficient, and low-cost** systems
for **widespread adoption** of data-intensive applications
in **healthcare** and **life sciences**
and **enabling new solutions** that were out of reach before*

- **Contact Info**

- <https://sites.google.com/view/nikamansourighiasi>
- n.mansourighiasi@gmail.com

Applications of (Meta)Genome Analysis

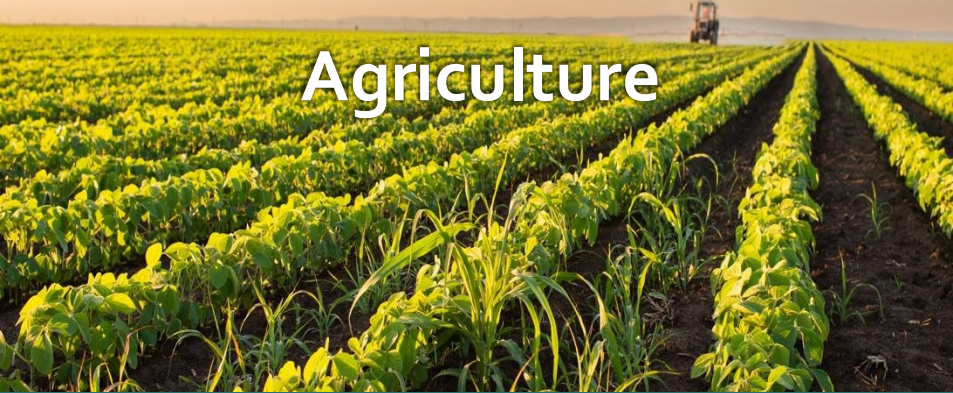
Precision Medicine



Outbreak Prevention and Management



Agriculture

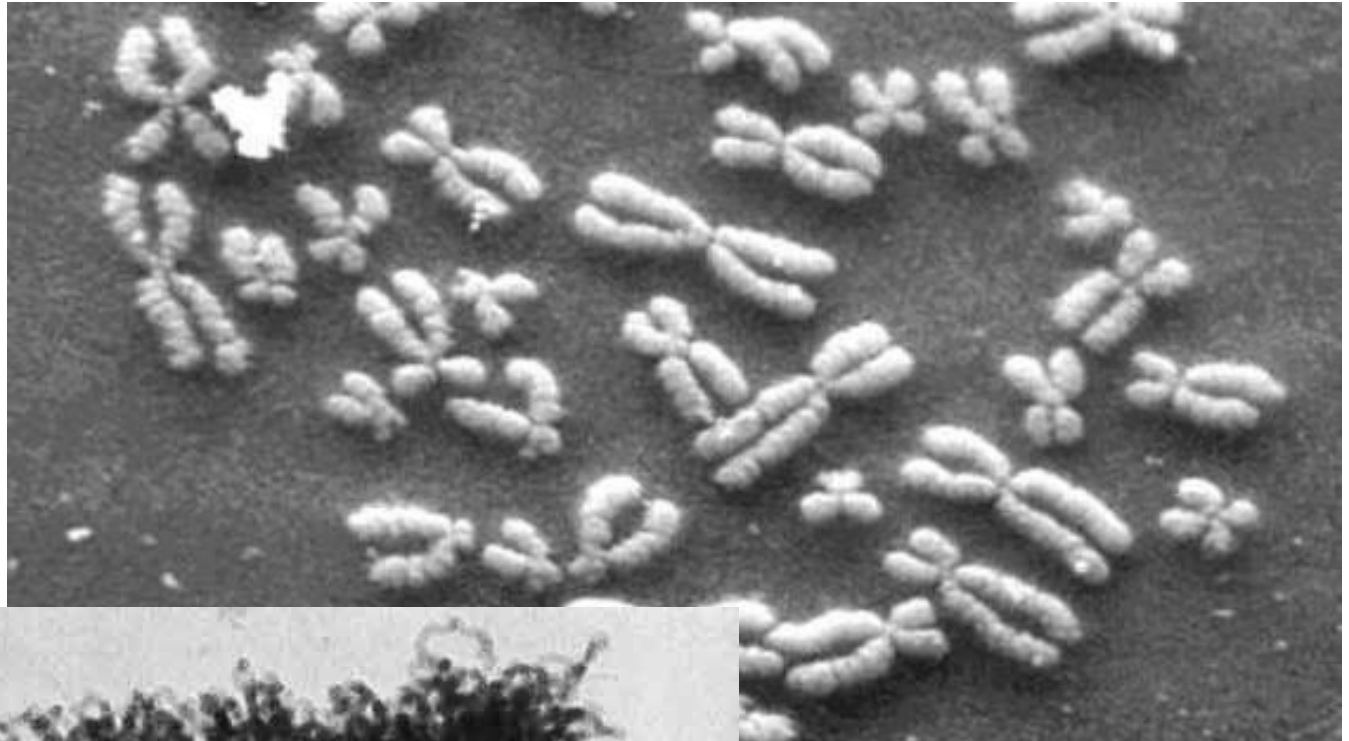
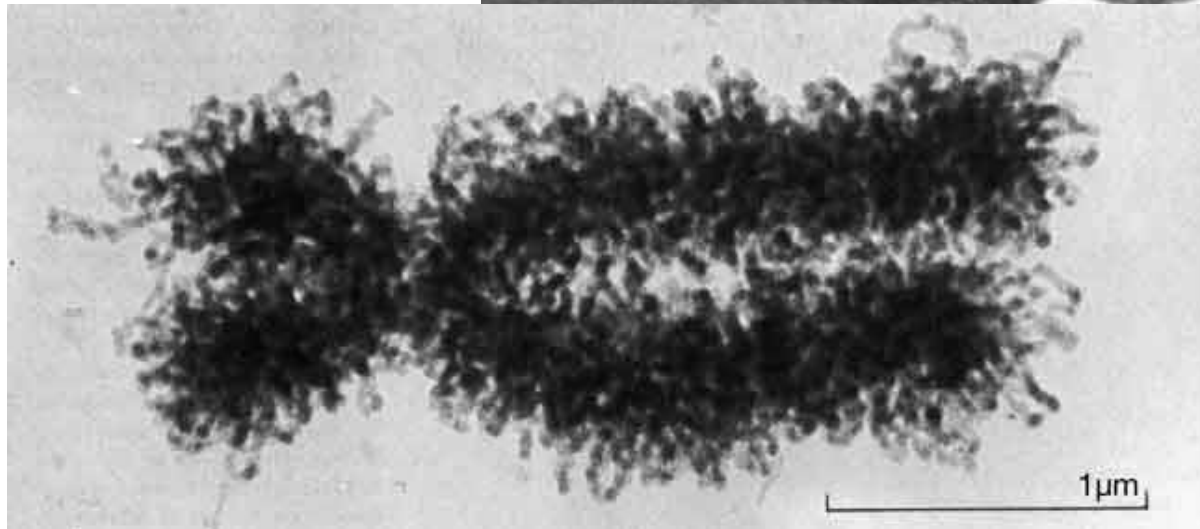


Scientific Discovery



& Many More...

DNA Under Electron Microscope



human chromosome #12
from HeLa cell

High-Throughput Sequencers



Illumina MiSeq



Pacific
Biosciences
Sequel II

Oxford
Nanopore
PromethION



Oxford
Nanopore
MinION



Illumina NovaSeq 6000



Pacific Biosciences RS II

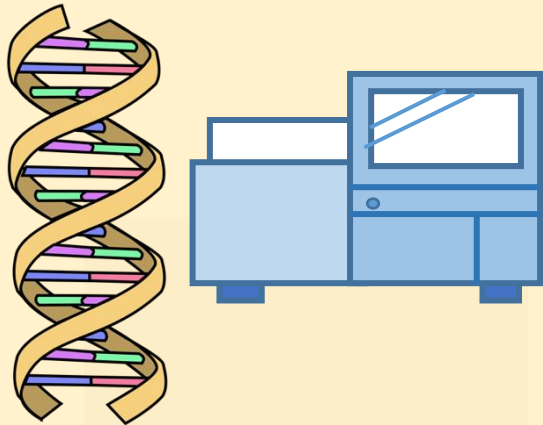


Oxford
Nanopore
GridION

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**



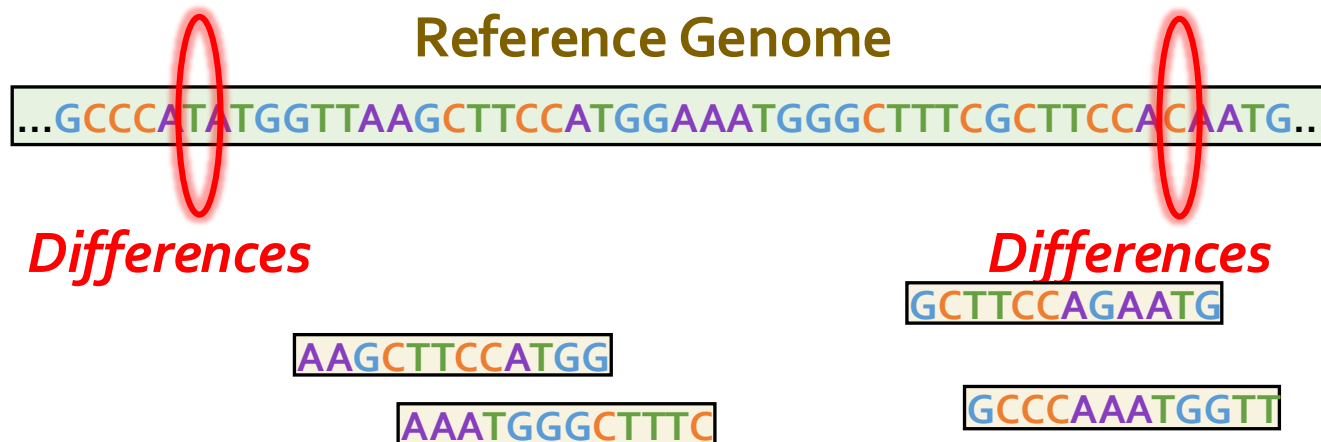
Illumina NovaSeq 6000

Pacific Biosciences RS II

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

Genome Sequence Analysis

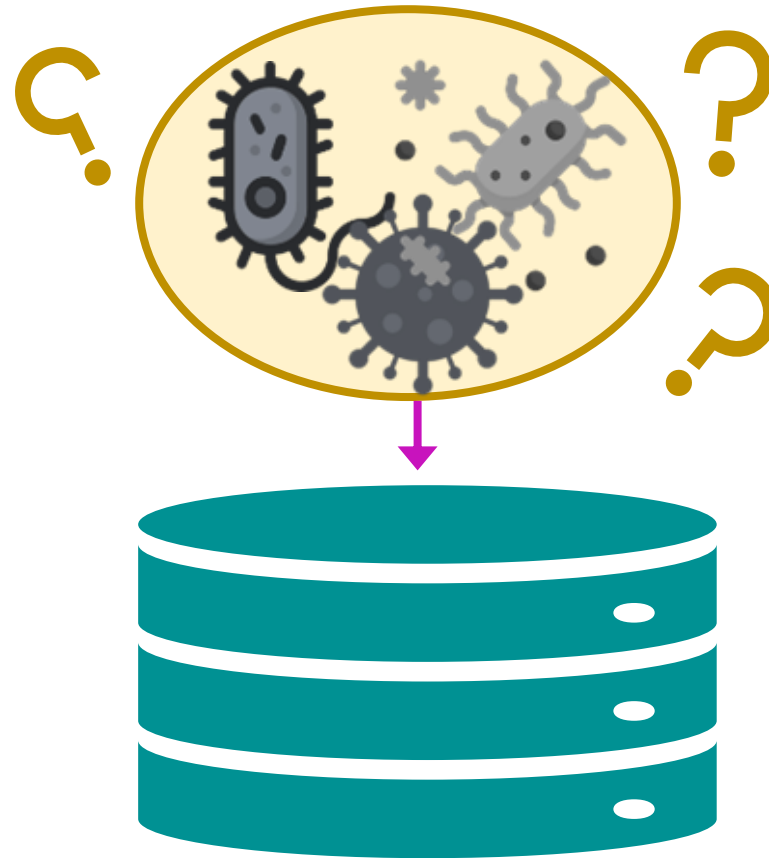
- **Read mapping:** first key step in genome sequence analysis
 - Aligns **reads** to potential **matching locations** in the **reference genome**
 - For each matching location, the **alignment step** finds the degree of **similarity (alignment score)**



- Calculating the alignment score requires **computationally-expensive approximate string matching (ASM)** to account for **differences** between reads and the reference genome

What is Metagenomics?

- **Metagenomics**: Study of genome sequences of **diverse organisms** within a **shared environment** (e.g., blood, ocean, soil)



A large database containing information
on **many species**

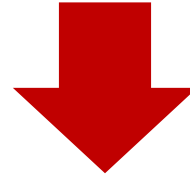
SAFARI

(e.g., > 100 TBs in emerging databases)

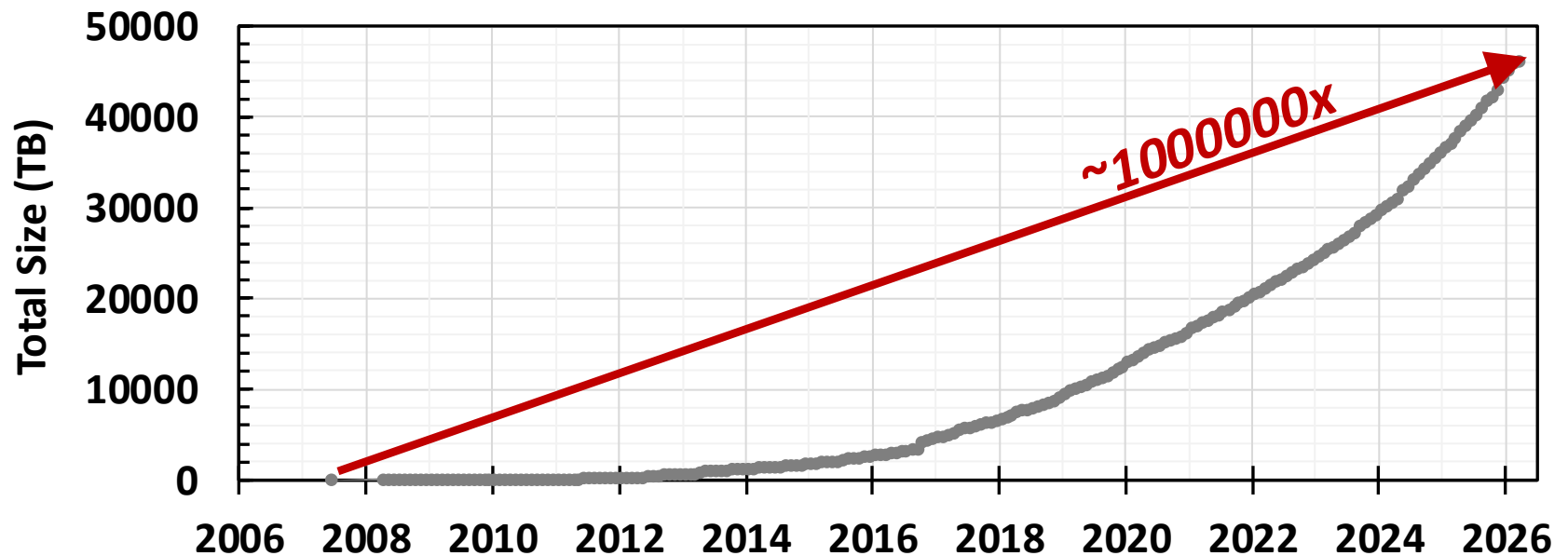
Sequence Data is Growing at a Fast Pace

Important Role
in Many Fields

Large Reduction
in Sequencing Cost

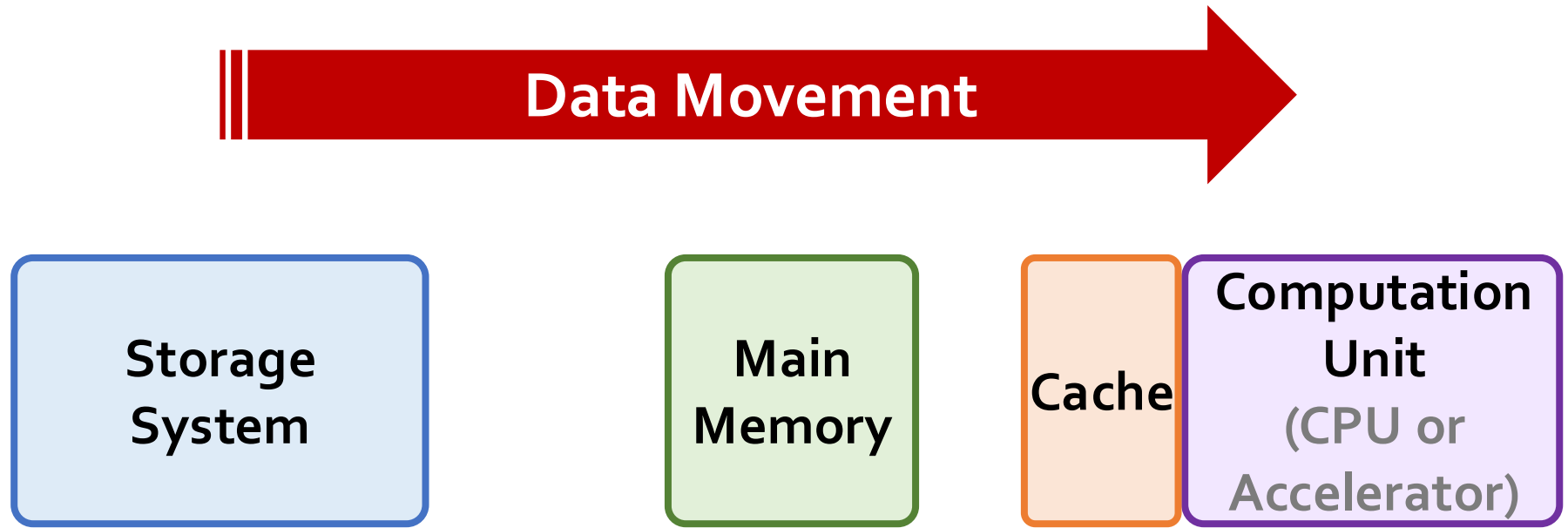


Fast growth in sequence data size



*Publicly-Hosted Genomic Sequence Reads Data Size
in Sequence Read Archive (SRA) Over Years*

Overheads of (Meta)Genomic Analyses

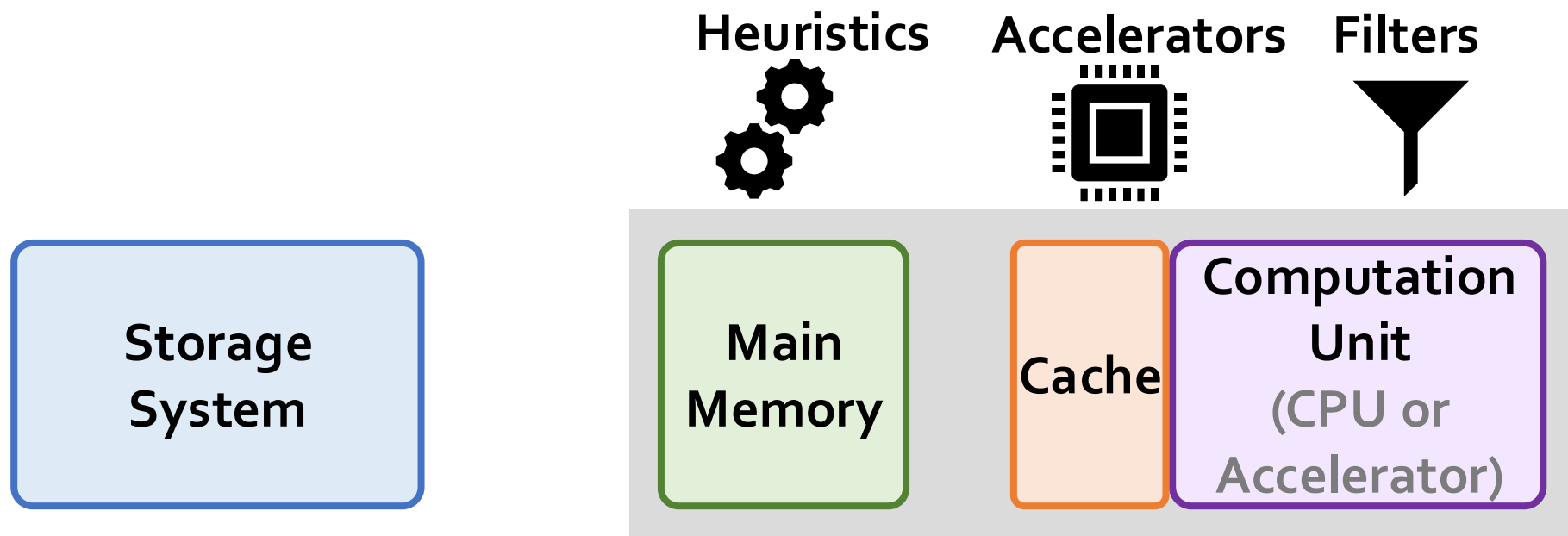


Computation overhead



Data movement overhead

Accelerating (Meta)Genome Sequence Analysis



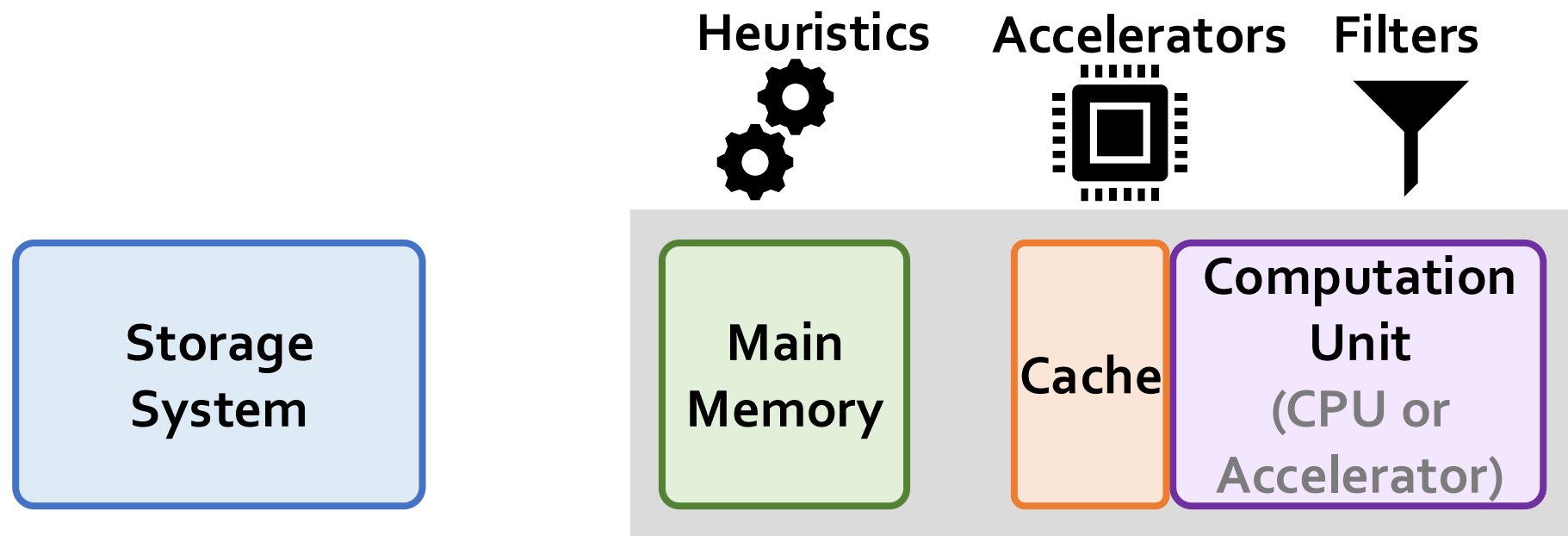
Computation overhead

Problems

Accessing stored sequence data and feeding it to the analysis units

Overhead of **moving large amounts of low-reuse data**
from the **storage system**
& **unnecessary computational burden** on the **rest of the system**

Accelerating (Meta)Genome Sequence Analysis



Computation overhead



Data movement overhead

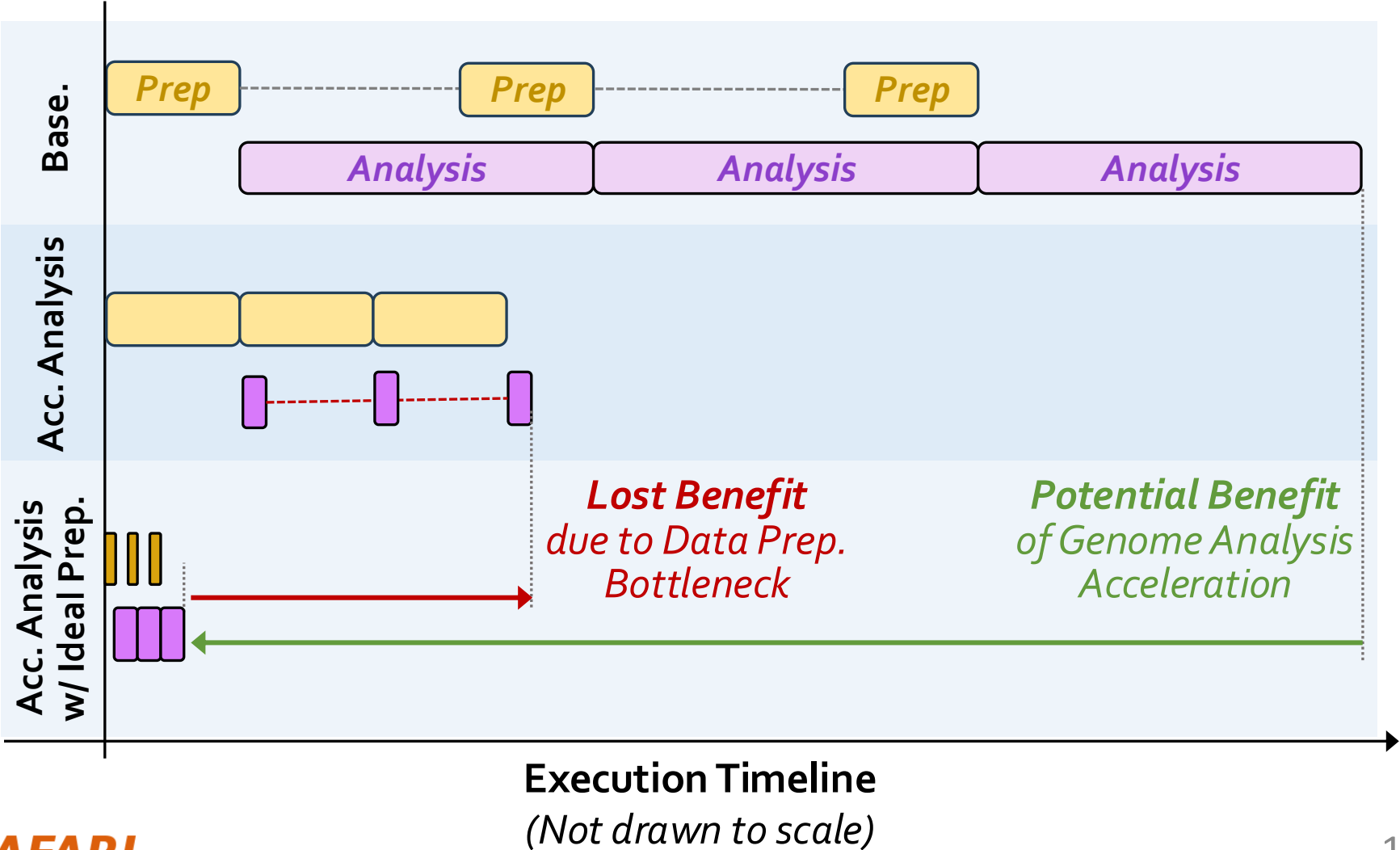
Problems

Accessing stored sequence data and feeding it to the analysis units

Overhead of **moving large amounts of low-reuse data**
from the **storage system**
& **unnecessary computational burden** on the **rest of the system**

Data preparation bottleneck,
where compressed genomic sequence data needs to
be first **decompressed** and **formatted** before it can be analyzed

Effect of the Data Preparation Bottleneck



Thesis Statement

By designing customized **storage-centric systems** that efficiently

① **analyze** genomic and metagenomic data **inside the storage system**

② enable **highly-compressed storage** and **high-performance access** of large-scale sequence data



we can alleviate the overheads of

data movement

computation

data preparation



thereby, significantly improving



performance



energy efficiency



cost effectiveness

Our Approach

Designing customized **storage-centric systems** that efficiently

① **analyze** genomic and metagenomic data **inside the storage system**

② enable **highly-compressed storage** and **high-performance access** of large-scale sequence data



we can alleviate the overheads of

data movement

computation

data preparation



thereby, significantly improving



performance



energy efficiency



cost effectiveness

Our Approach

Designing customized *storage-centric systems* that efficiently

① *analyze* genomic and metagenomic data *inside the storage system*

① **GenStore** [ASPLOS'22]

In-storage filtering
for **genomic** analysis

② **MegIS** [ISCA'24]

Cooperative in-storage processing
for **metagenomic** analysis

③ **GRAINS** [ISCA'26]

Storage-aware algorithm-architecture co-design
for **graph-based** (meta)genomic analyses

② *enable highly-compressed storage and high-performance access
of large-scale sequence data*

④ **SAGe** [HPCA'26]

Algorithm-architecture co-design
for **storing** and **accessing** sequence data

Our Approach

Designing customized **storage-centric systems** that efficiently

① **analyze** genomic and metagenomic data **inside the storage system**

① **GenStore** [ASPLOS'22]

In-storage filtering
for **genomic** analysis

② **MegIS** [ISCA'24]

Cooperative in-storage processing
for **metagenomic** analysis

③ **GRAINS** [ISCA'26]

Storage-aware algorithm-architecture co-design
for **graph-based** (meta)genomic analyses

② **enable highly-compressed storage and high-performance access**
of large-scale sequence data

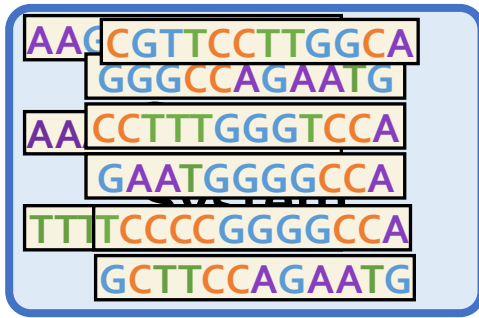
④ **SAGe** [HPCA'26]

Algorithm-architecture co-design
for **storing and accessing** sequence data

Key Idea



*Filter reads that do **not** require alignment inside the storage system*



Filtered Reads

**Main
Memory**

Cache

**Computation
Unit**
(CPU or
Accelerator)

Exactly-matching reads

Do not need expensive approximate string matching during alignment

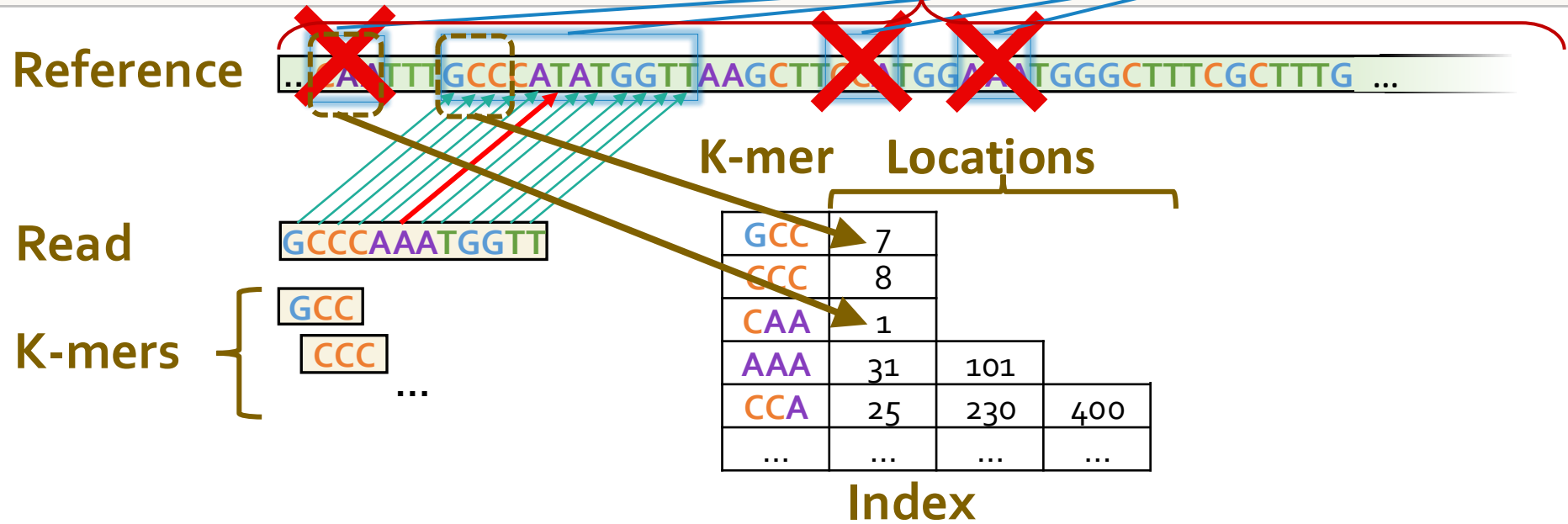
Non-matching reads

Do not have potential matching locations and can skip alignment

Read Mapping Process

> 3 billion characters
for the human genome

Seed Hits



Seeding	Determine potential matching locations (seed hits) in the reference genome
Seed Filtering (e.g., Chaining)	Prune some seed hits in the reference genome
Alignment	Determine the exact differences between the read and the reference genome

GenStore for Exactly-Matching Reads

GenStore for Non-Matching Reads

GenStore for Exactly-Matching Reads

GenStore for Non-Matching Reads

GenStore for Exactly-Matching Reads

- Efficient in-storage filter for reads with at least one **exact match** in the reference genome
- Uses **simple operations**, without requiring alignment
- **Challenge:** Large number of **random accesses per read** to the reference genome and its index

Expensive random accesses to flash chips

Limited DRAM capacity inside the SSD

Data Structures

- **Read-sized k-mers:** to reduce the number of accesses per each read



- **Sorted read-sized k-mers:** to avoid random accesses to the index

✓ Sequential scan of the read set and the index

Index Optimization

- Read-sized k-mer index takes up a **large amount of space** (126 GB for human index) due to the larger number of unique k-mers

Sorted K-mer Index

Strong Hash Value	Loc.
1	1, 8, ...
4	51
7	23, 37
...	...

Using strong hash values instead of read-sized k-mers
reduces the size of the index by 3.9x

Evaluation Methodology

Read Mappers

- **Base:** state-of-the-art software or hardware read mappers
 - **Minimap2** [Bioinformatics'18]: software mapper for **short and long reads**
 - **GenCache** [MICRO'19]: hardware mapper for **short reads**
 - **Darwin** [ASPLOS'18]: hardware mapper for **long reads**
- **GS:** Base integrated with GenStore

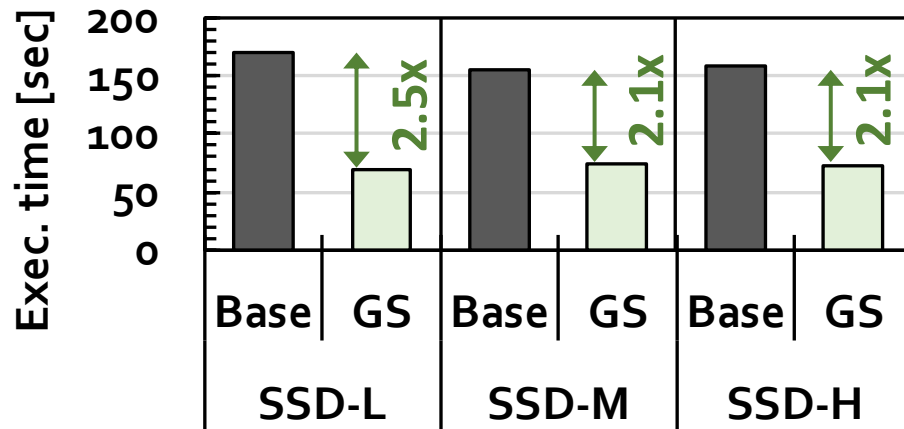
SSD Configurations

- **SSD-L:** With **SATA3** interface (**0.5 GB/s** sequential read bandwidth)
- **SSD-M:** With **PCIe Gen3** interface (**3.5 GB/s** sequential read bandwidth)
- **SSD-H:** With **PCIe Gen4** interface (**7 GB/s** sequential read bandwidth)

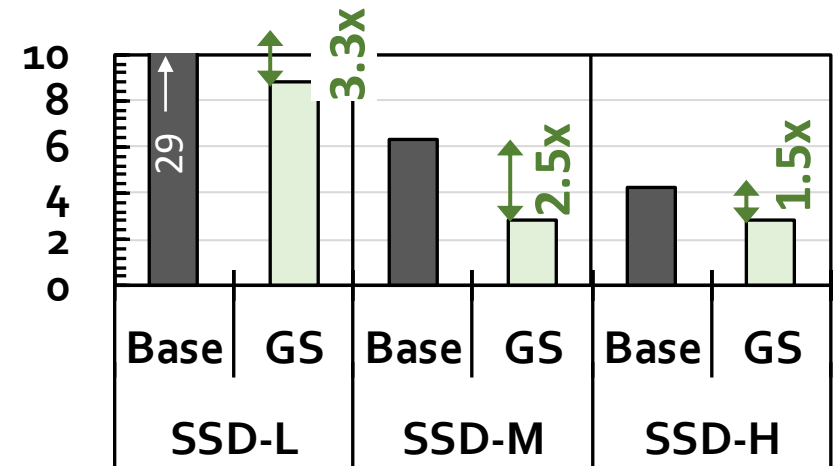
GenStore for Exactly-Matching Reads

For a read set with 80% exactly-matching reads

With the Software Mapper



With the Hardware Mapper



2.1x - 2.5x speedup compared to the software Base

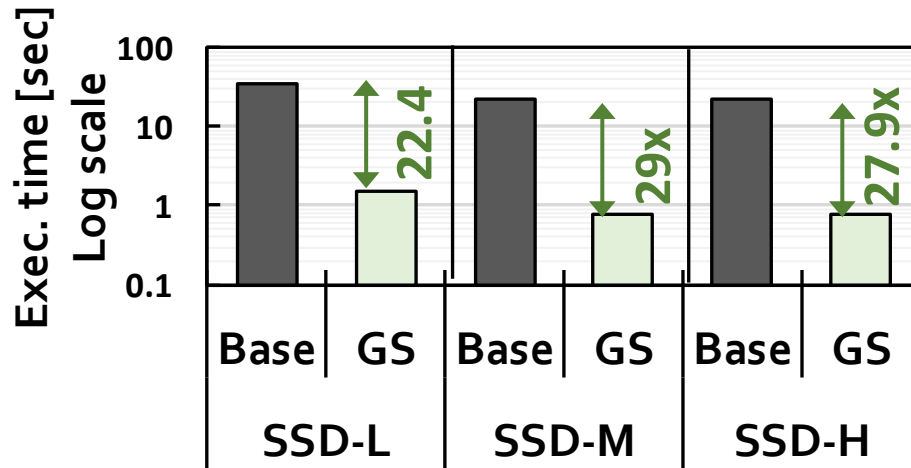
1.5x – 3.3x speedup compared to the hardware Base

On average 3.92x energy reduction

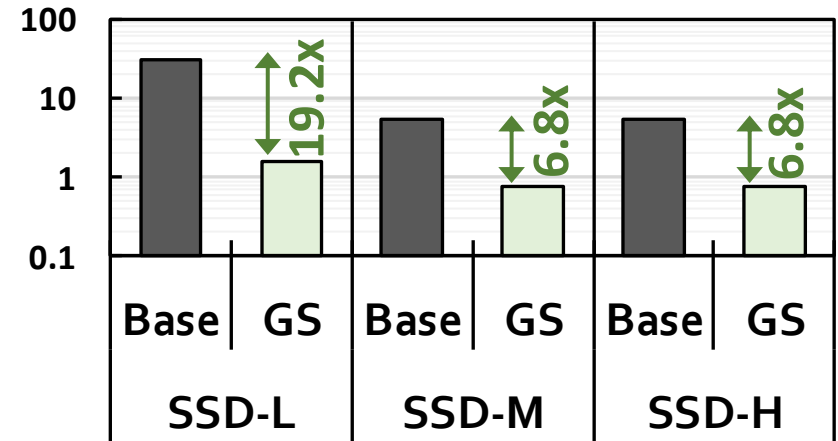
GenStore for Non-Matching Reads

For a read set with 99.7% non-matching reads

With the Software Mapper



With the Hardware Mapper



22.4x – 27.9x speedup compared to the software Base

6.8x – 19.2x speedup compared to the hardware Base

On average 27.2x energy reduction

Area and Power

- Based on **Synthesis** of **GenStore** accelerators using the Synopsys Design Compiler @ 65nm technology node

Logic unit	# of instances	Area [mm ²]	Power [mW]
Comparator	1 per SSD	0.0007	0.14
K -mer Window	2 per channel	0.0018	0.27
Hash Accelerator	2 per SSD	0.008	1.8
Location Buffer	1 per channel	0.00725	0.37375
Chaining Buffer	1 per channel	0.008	0.95
Chaining PE	1 per channel	0.004	0.98
Control	1 per SSD	0.0002	0.11
<i>Total for an 8-channel SSD</i>	-	0.2	26.6

Only 0.006% of a 14nm Intel Processor,

less than 9.5% of the three ARM processors in a SATA SSD controller

Our Approach

Designing customized **storage-centric systems** that efficiently

① **analyze** genomic and metagenomic data **inside the storage system**

① **GenStore** [ASPLOS'22]

In-storage filtering
for **genomic** analysis

② **MegIS** [ISCA'24]

Cooperative in-storage processing
for **metagenomic** analysis

③ **GRAINS** [ISCA'26]

Storage-aware algorithm-architecture co-design
for **graph-based** (meta)genomic analyses

② **enable highly-compressed storage and high-performance access**
of large-scale sequence data

④ **SAGe** [HPCA'26]

Algorithm-architecture co-design
for **storing and accessing** sequence data

Our Approach

Designing customized **storage-centric systems** that efficiently

① **analyze** genomic and metagenomic data **inside the storage system**

① **GenStore** [ASPLOS'22]

In-storage filtering
for **genomic** analysis

② **MegIS** [ISCA'24]

Cooperative in-storage processing
for **metagenomic** analysis

③ **GRAINS** [ISCA'26]

Storage-aware algorithm-architecture co-design
for **graph-based** (meta)genomic analyses

② **enable highly-compressed storage and high-performance access**
of large-scale sequence data

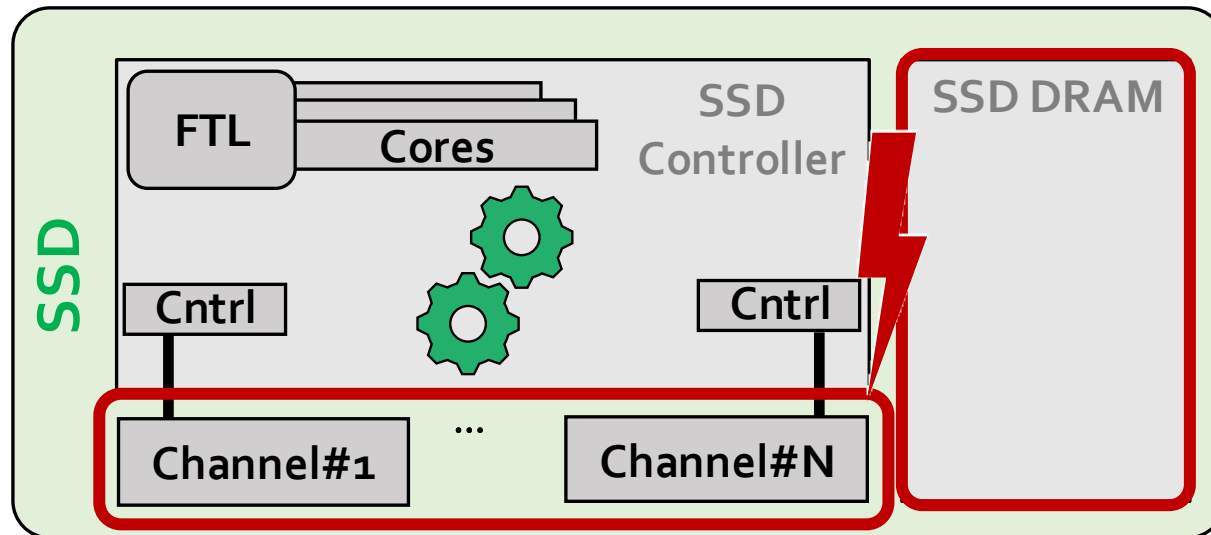
④ **SAGe** [HPCA'26]

Algorithm-architecture co-design
for **storing and accessing** sequence data

Challenges of Storage-Centric Computing

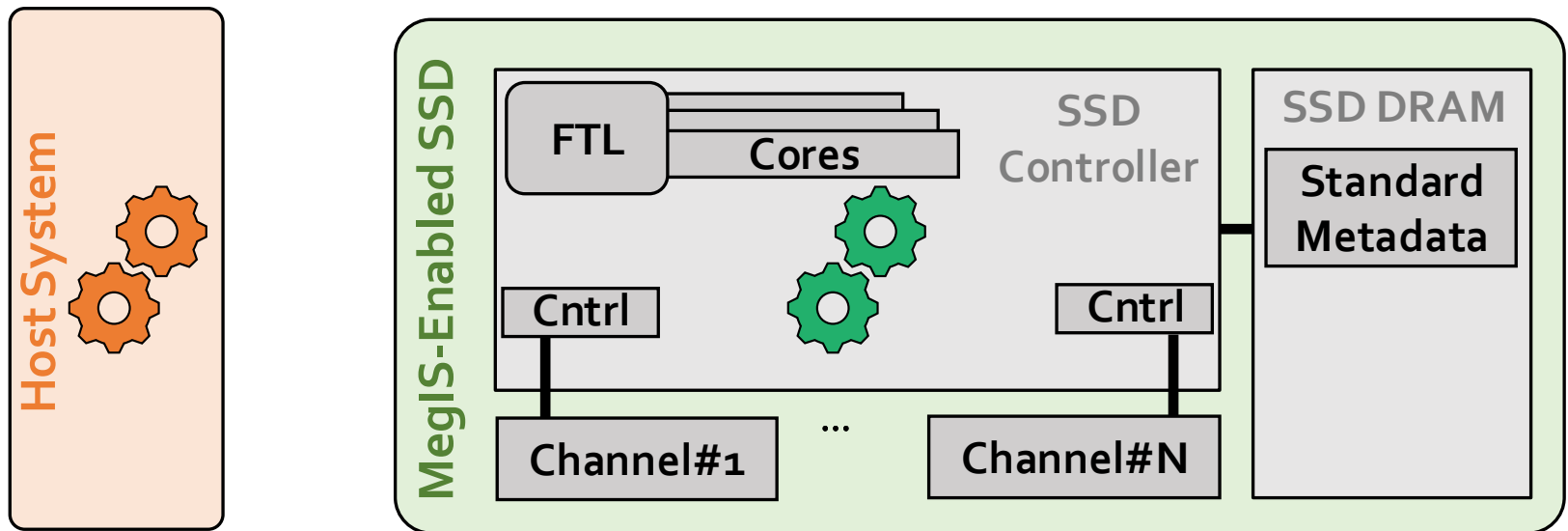
Metagenomic analysis tools cannot run in-storage due to SSD limits

- Long **latency of NAND flash** chips
- Limited **DRAM capacity** inside the SSD
- Limited **DRAM bandwidth** inside the SSD

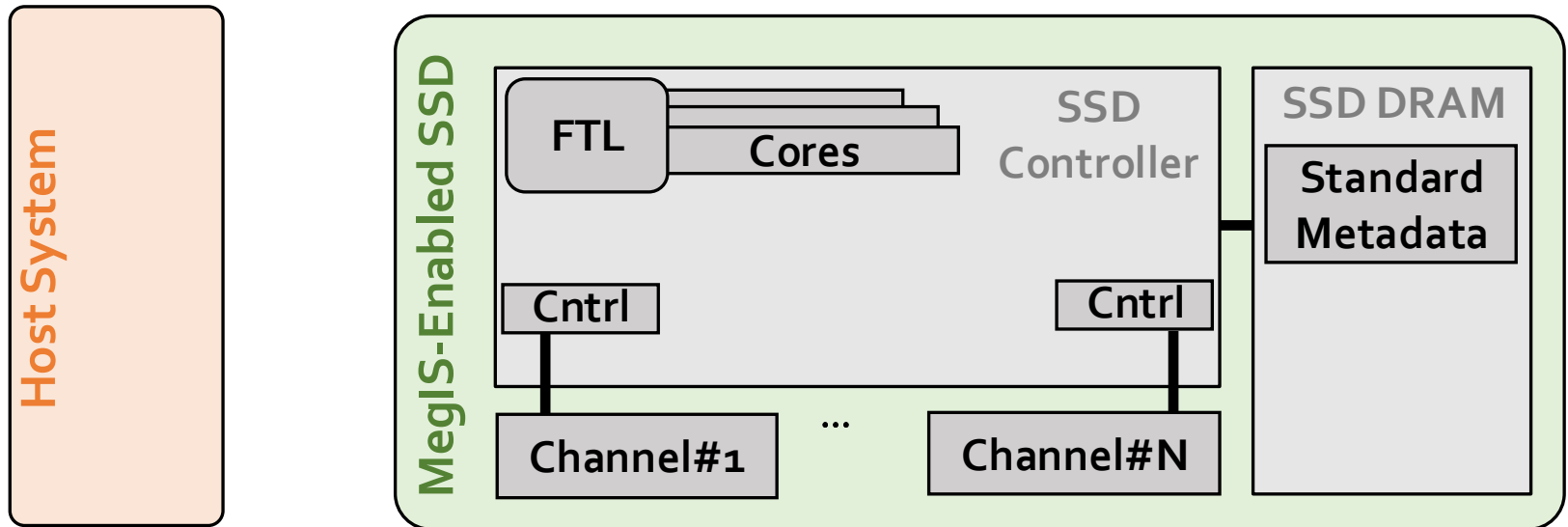


MegIS: Metagenomics In-Storage

- First in-storage system for *end-to-end* metagenomic analysis
- **Idea:** Cooperative in-storage processing for metagenomic analysis
 - Algorithm-Architecture co-design between the



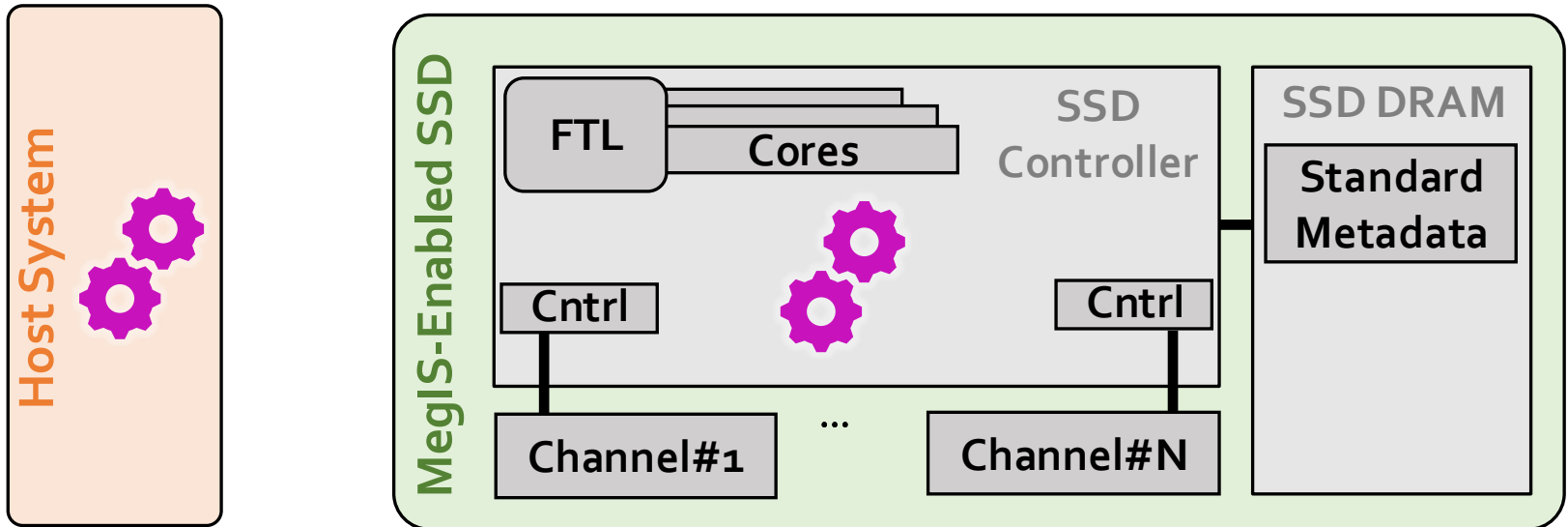
MegIS Algorithm-Architecture Co-Design



MegIS Algorithm-Architecture Co-Design

Task partitioning and mapping

- *Each step executes in its most suitable system*



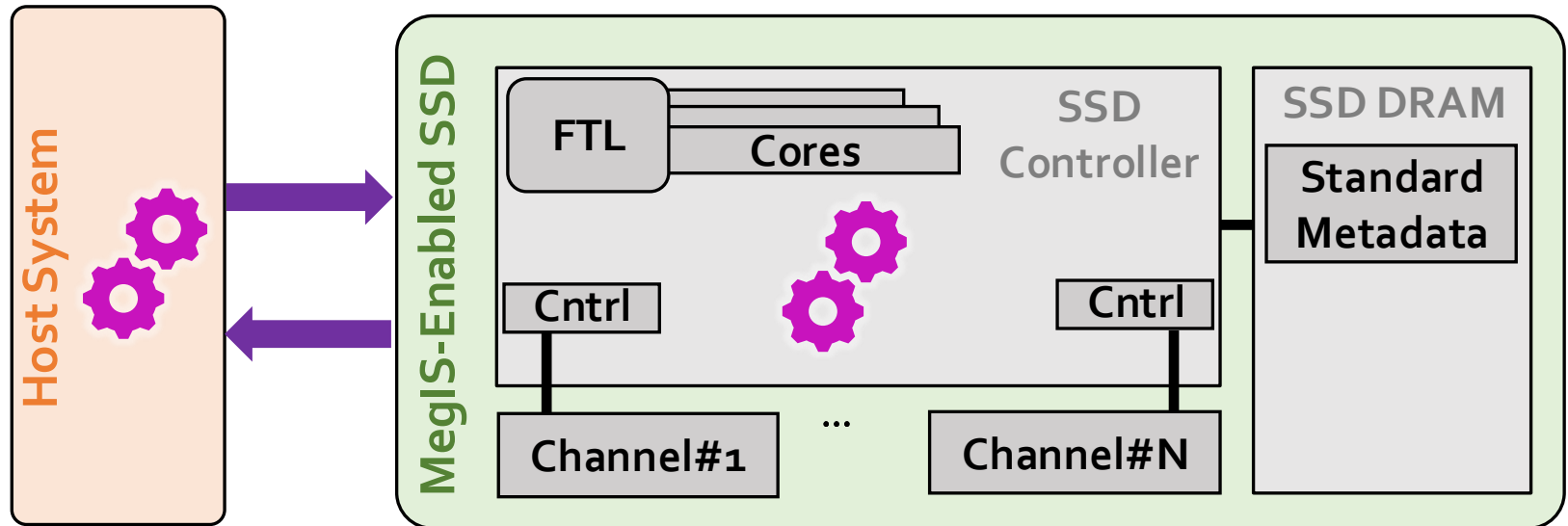
MegIS Algorithm-Architecture Co-Design

Task partitioning and mapping

- *Each step executes in its most suitable system*

Data/computation flow coordination

- *Reduce communication overhead*
- *Reduce #writes to flash chips*



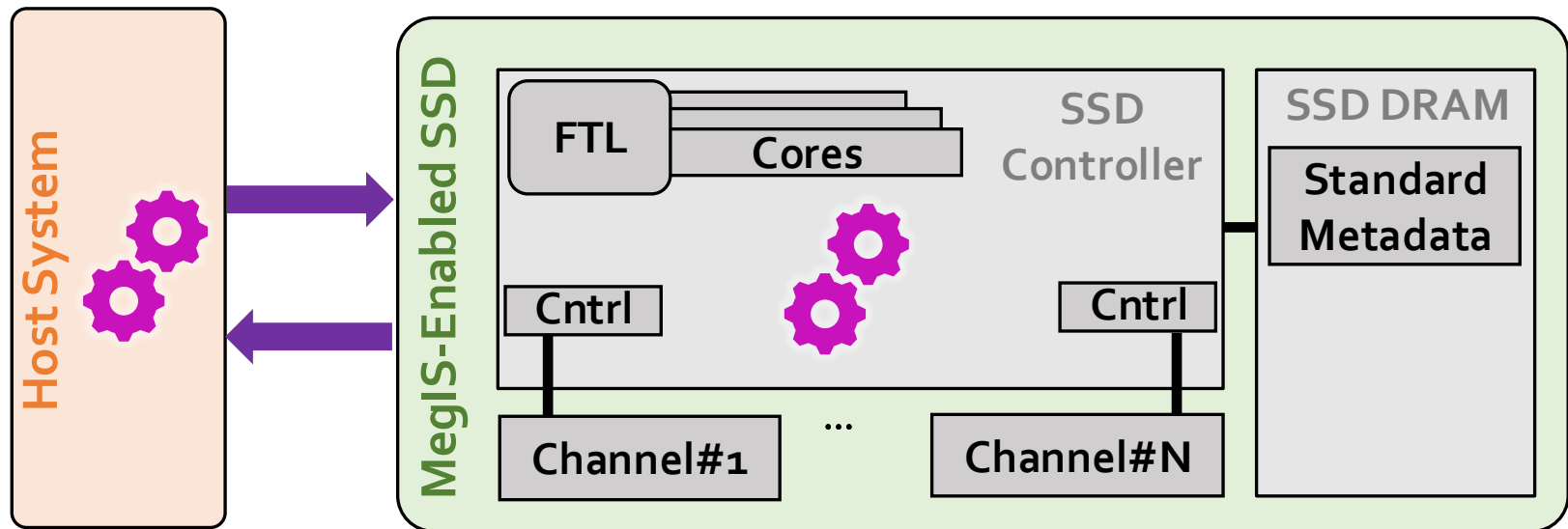
MegIS Algorithm-Architecture Co-Design

Task partitioning and mapping

- *Each step executes in its most suitable system*

Data/computation flow coordination

- *Reduce communication overhead*
- *Reduce #writes to flash chips*



Storage-aware algorithms

- *Enable efficient access patterns to the SSD*

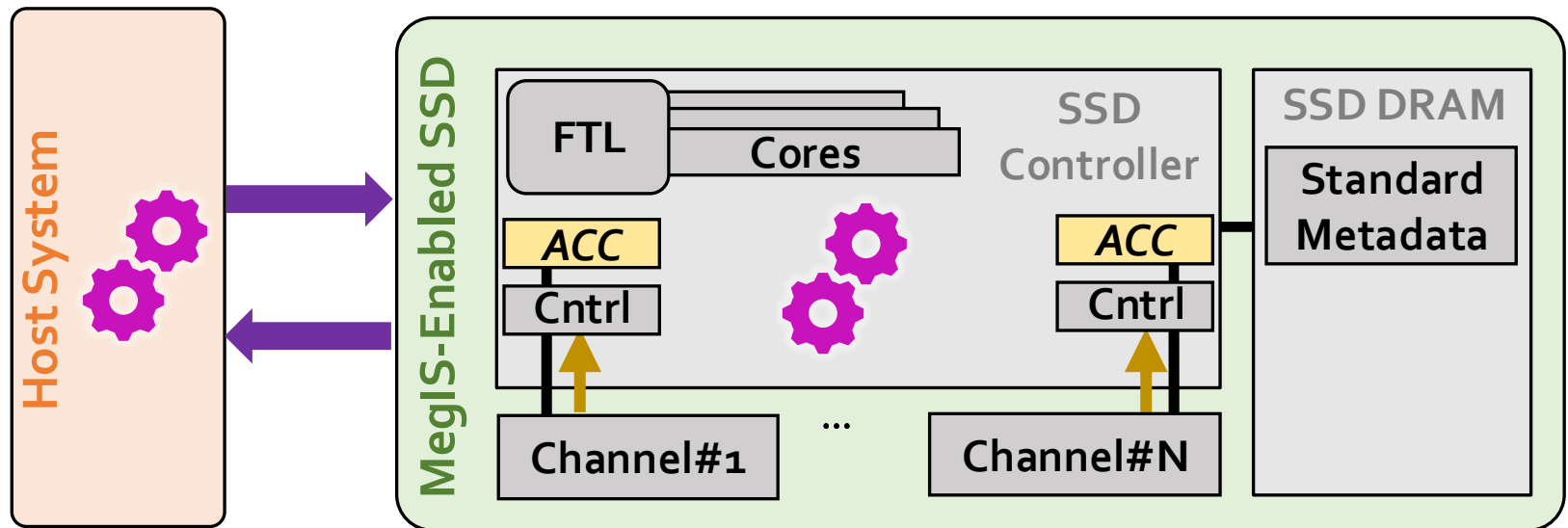
MegIS Algorithm-Architecture Co-Design

Task partitioning and mapping

- *Each step executes in its most suitable system*

Data/computation flow coordination

- *Reduce communication overhead*
- *Reduce #writes to flash chips*



Storage-aware algorithms

- *Enable efficient access patterns to the SSD*

Lightweight in-storage accelerators

- *Minimize SRAM/DRAM buffer spaces needed inside the SSD*

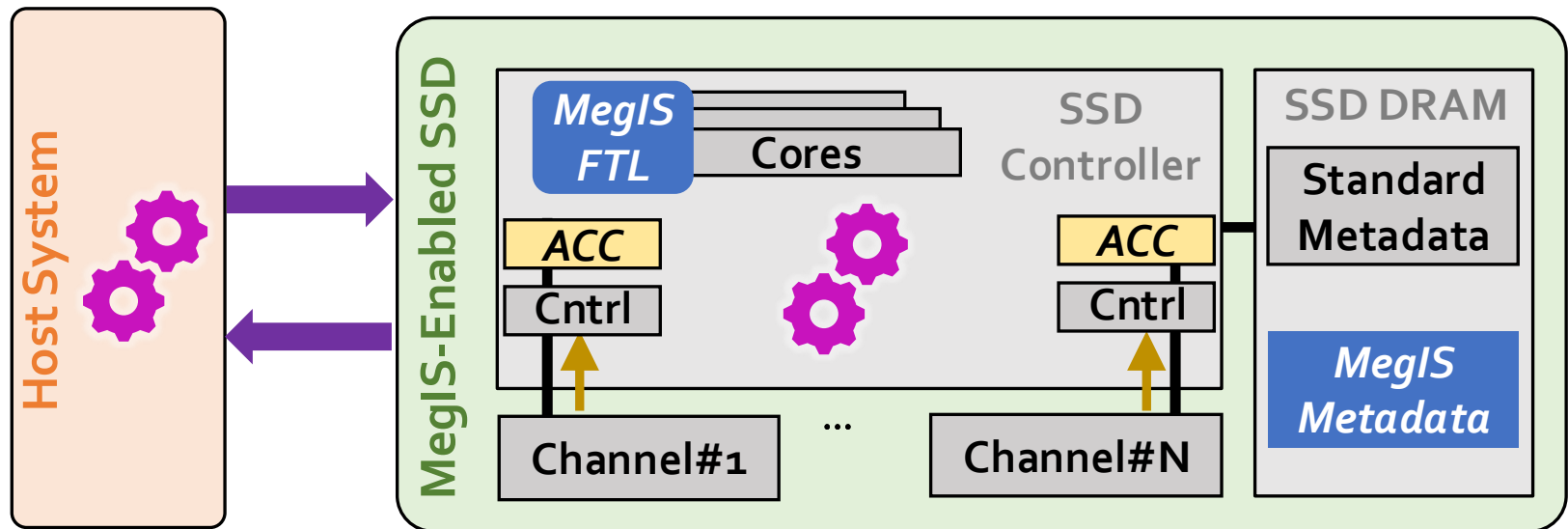
MegIS Algorithm-Architecture Co-Design

Task partitioning and mapping

- Each step executes in its most suitable system

Data/computation flow coordination

- Reduce communication overhead
- Reduce #writes to flash chips



Storage-aware algorithms

- Enable efficient access patterns to the SSD

Lightweight in-storage accelerators

- Minimize SRAM/DRAM buffer spaces needed inside the SSD

Data mapping scheme and Flash Translation Layer (FTL)

- Specialize to the characteristics of metagenomic analysis
- Leverage the SSD's full internal bandwidth

Evaluation Methodology Overview (I)

Performance, Energy, and Power Analysis

Hardware Components

- Synthesized Verilog model for the in-storage accelerators
- 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® host CPU with 128 physical cores
- 1 TB host DRAM

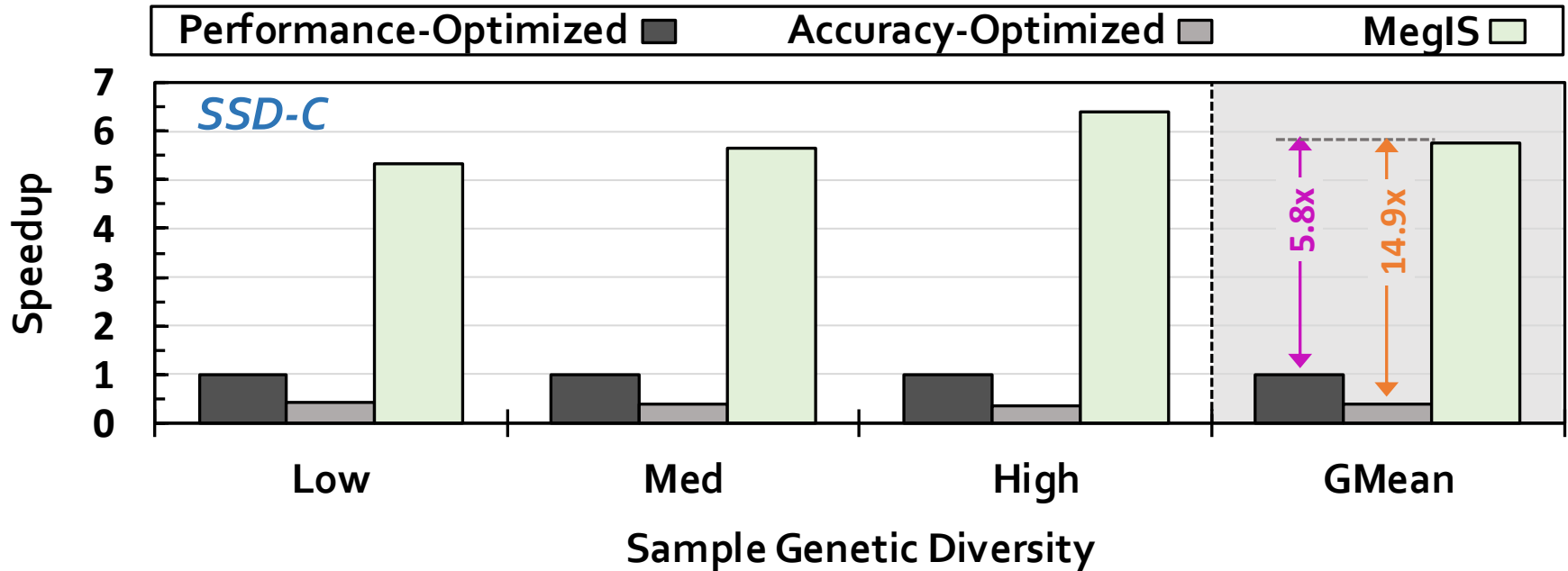
Baseline Comparison Points

- **Performance-optimized software**, Kraken2 [Genome Biology'19]
- **Accuracy-optimized software**, Metalign [Genome Biology'20]
- **PIM hardware-accelerated tool** (using processing-in-memory), Sieve [ISCA'21]

SSD Configurations

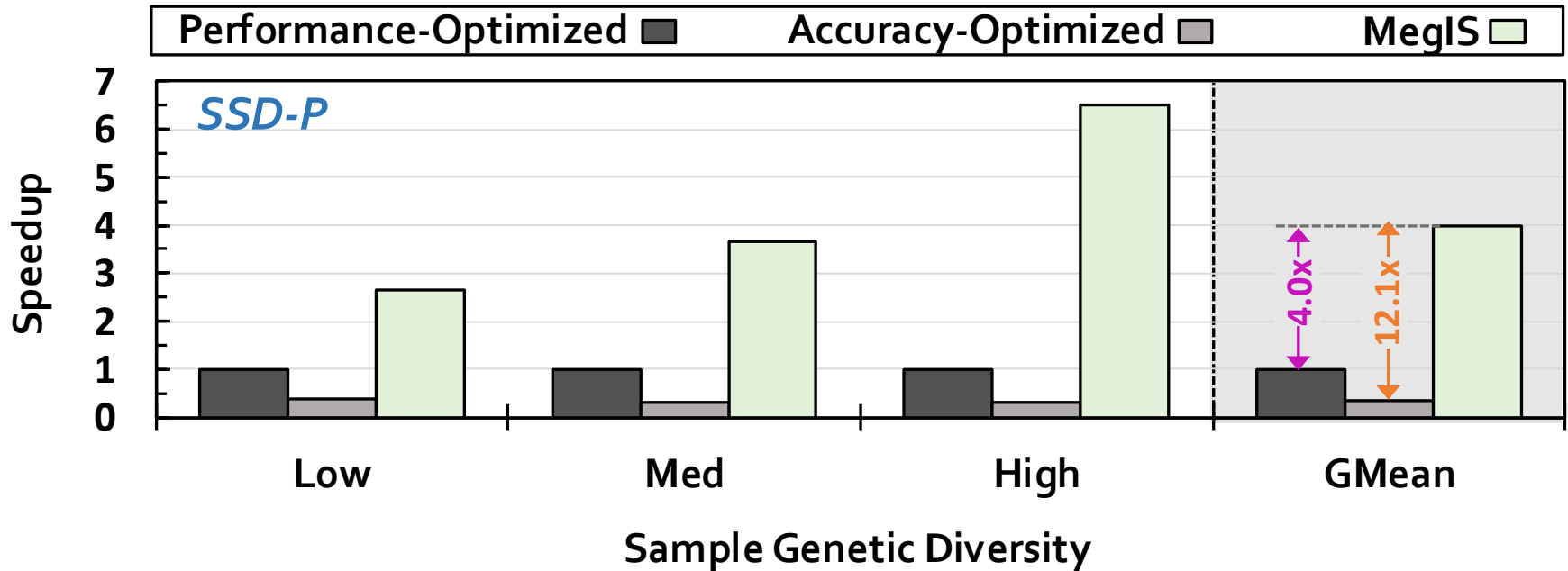
- **SSD-C:** With SATA3 interface (0.5 GB/s sequential read bandwidth)
- **SSD-P:** With PCIe Gen4 interface (7 GB/s sequential read bandwidth)

Speedup over Software (with Cost-Optimized SSD)



MegIS provides significant speedup over both
Performance-Optimized and **Accuracy-Optimized** baselines

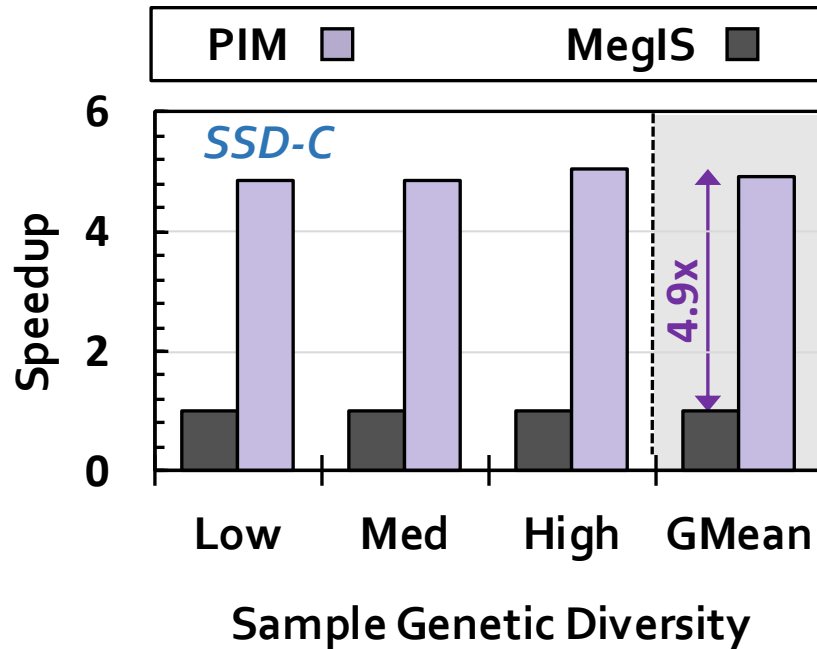
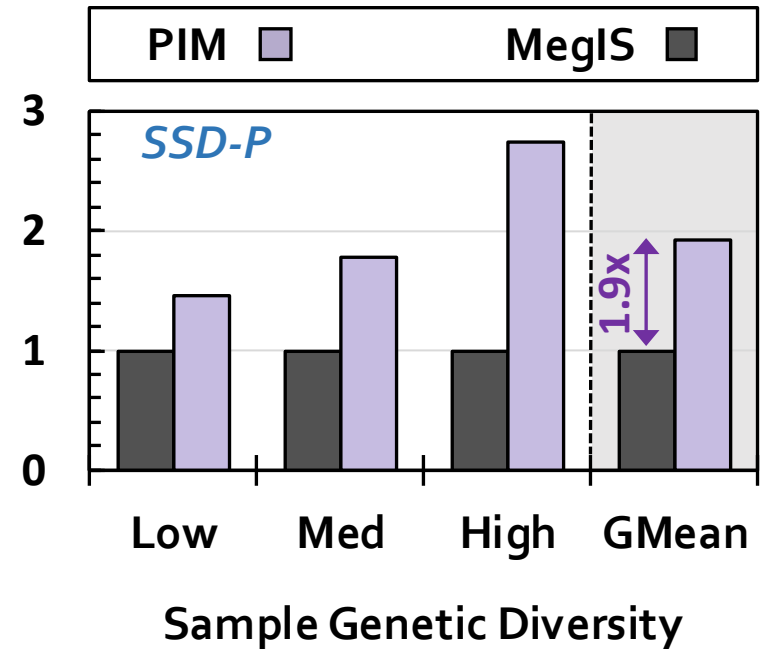
Speedup over Software (with Performance-Optimized SSD)



MegIS provides significant speedup over both
Performance-Optimized and **Accuracy-Optimized** baselines

MegIS improves performance on both
cost-optimized and performance-optimized SSDs

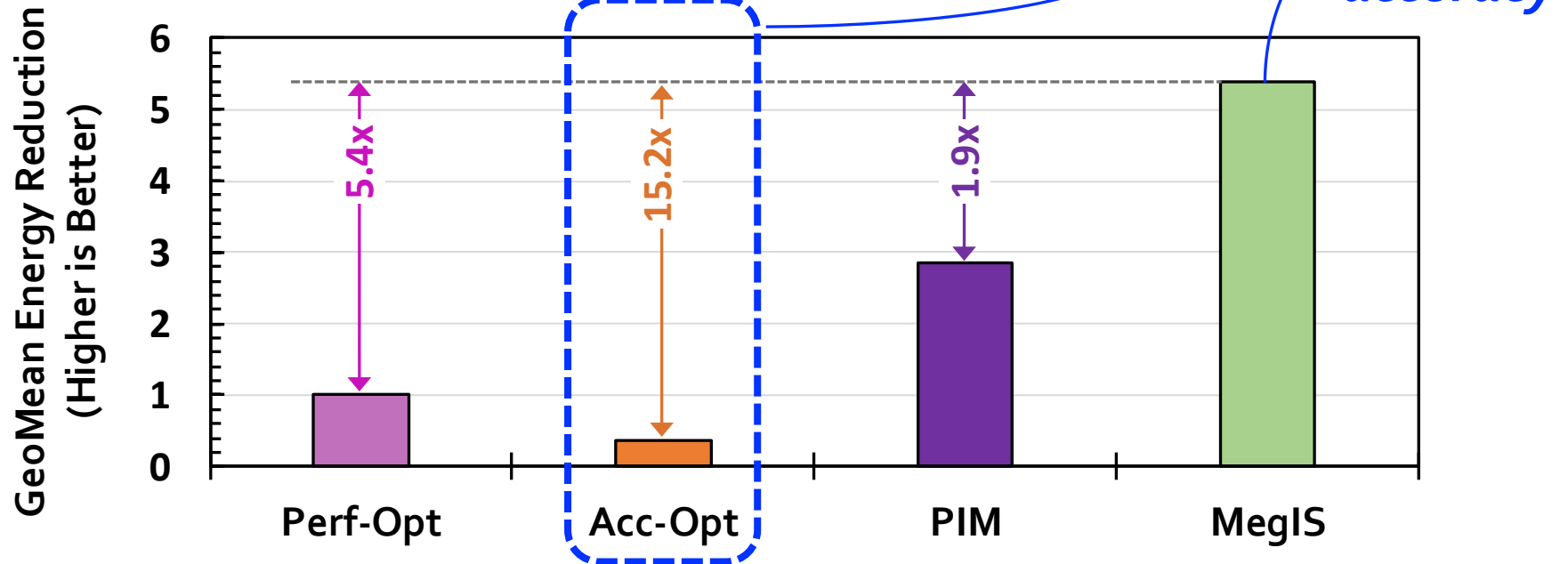
Speedup over the PIM Hardware Baseline



MegIS provides significant speedup over the **PIM** baseline

Reduction in Energy Consumption

- On average across different input sets and SSDs



MegIS provides significant energy reduction over the **Performance-Optimized**, **Accuracy-Optimized**, and **PIM** baselines

Accuracy, Area, and Power

Accuracy

- **Same accuracy** as the **accuracy-optimized** baseline
- **Significantly higher accuracy** than the **performance-optimized** and **PIM** baselines
 - 4.6 – 5.2× higher F1 score
 - 3 – 24% lower L1 norm error

Area and Power

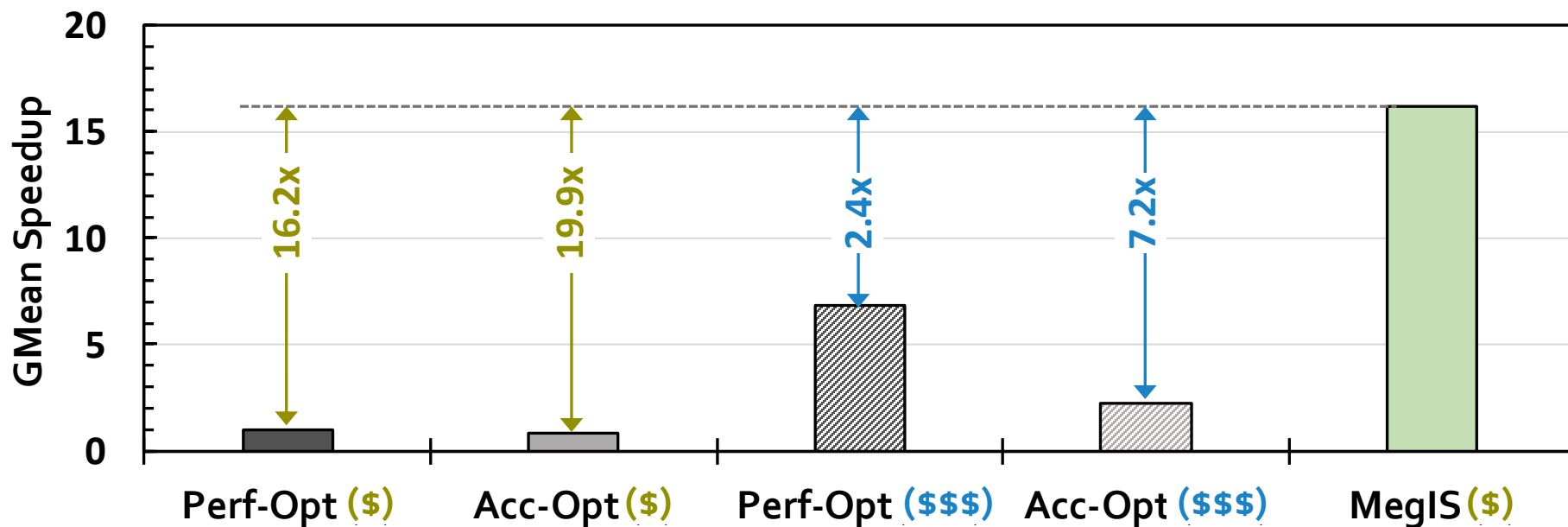
Total for an 8-channel SSD:

- **Area:** 0.04 mm²
- **Power:** 7.658 mW

*(Only **1.7%** of the area and **4.6%** of the power consumption of three ARM Cortex R4 cores in an SSD controller)*

System Cost-Efficiency

- **Cost-optimized system (\$):** With SSD-C and 64-GB DRAM
- **Performance-optimized system (\$\$\$):** With SSD-P and 1-TB DRAM



**MegIS outperforms the baselines
even when running on a much less costly system**

Our Approach

Designing customized **storage-centric systems** that efficiently

① **analyze** genomic and metagenomic data **inside the storage system**

① **GenStore** [ASPLOS'22]

In-storage filtering
for **genomic** analysis

② **MegIS** [ISCA'24]

Cooperative in-storage processing
for **metagenomic** analysis

③ **GRAINS** [ISCA'26]

Storage-aware algorithm-architecture co-design
for **graph-based** (meta)genomic analyses

② **enable highly-compressed storage and high-performance access**
of large-scale sequence data

④ **SAGe** [HPCA'26]

Algorithm-architecture co-design
for **storing and accessing** sequence data

Our Approach

Designing customized **storage-centric systems** that efficiently

① **analyze** genomic and metagenomic data **inside the storage system**

1 **GenStore** [ASPLOS'22]
In-storage filtering
for **genomic** analysis

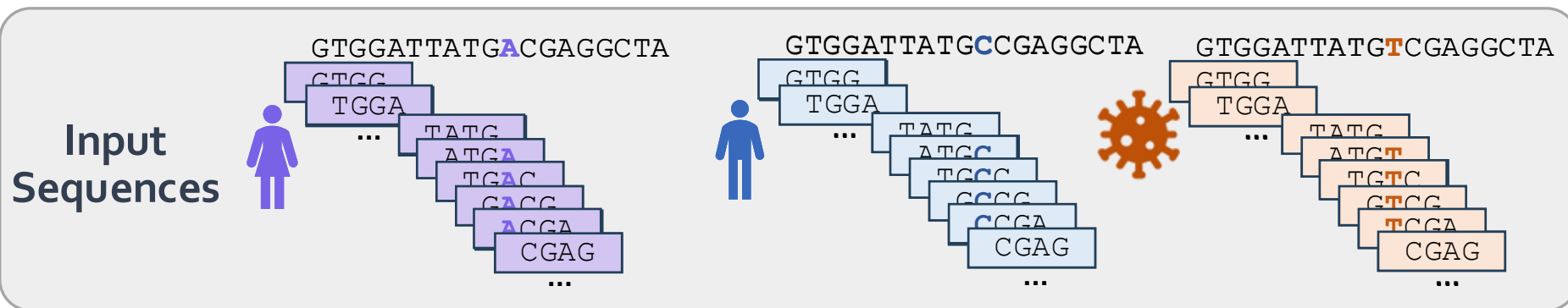
2 **MegIS** [ISCA'24]
Cooperative in-storage processing
for **metagenomic** analysis

3 **GRAINS** [ISCA'26]
Storage-aware algorithm-architecture co-design
for **graph-based** (meta)genomic analyses

② **enable highly-compressed storage and high-performance access**
of large-scale sequence data

4 **SAGe** [HPCA'26]
Algorithm-architecture co-design
for **storing and accessing** sequence data

Large-Scale Genome Graphs



Graph Representation



Powerful and efficient representations of genomic data

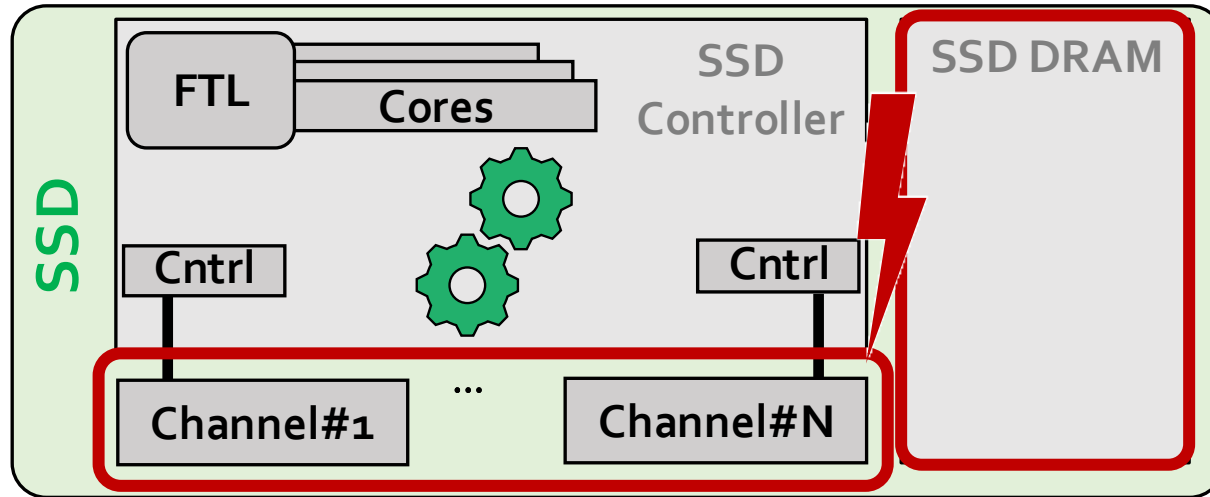
✓ Great **expressive** power

✓ **Compact** representations

Critical for enabling **massive, population-scale analyses**

Challenges of Storage-Centric Computing

Graph-based sequence analysis tools cannot run in storage due to SSD limits



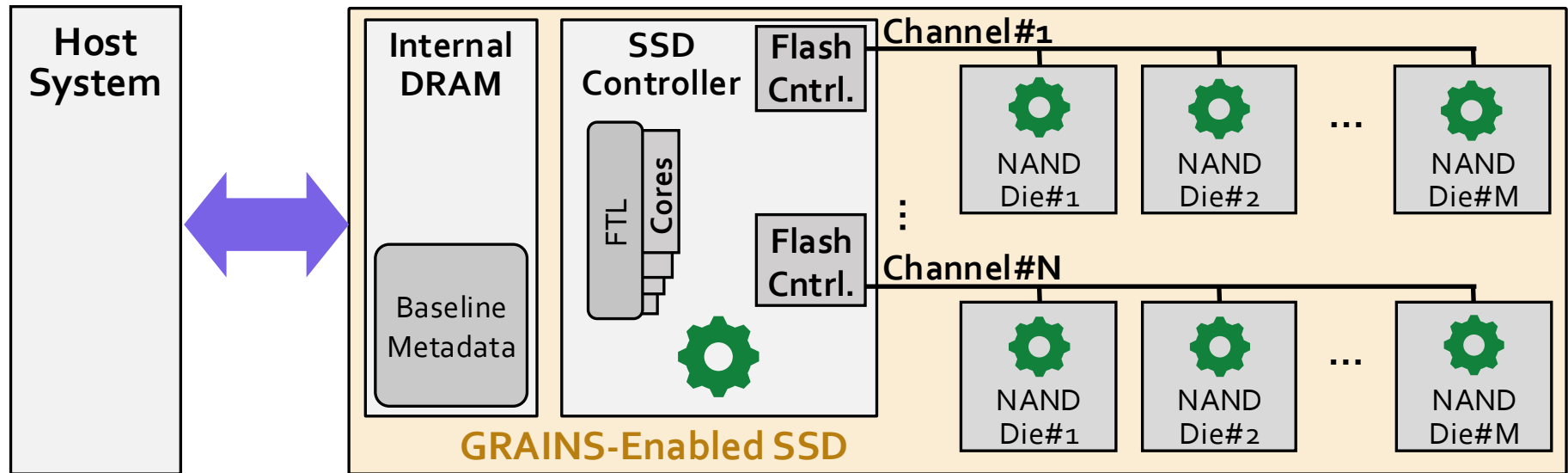
New challenges due to:

Dependent and random accesses to the sequence graph

Different properties from conventional, non-genome graphs

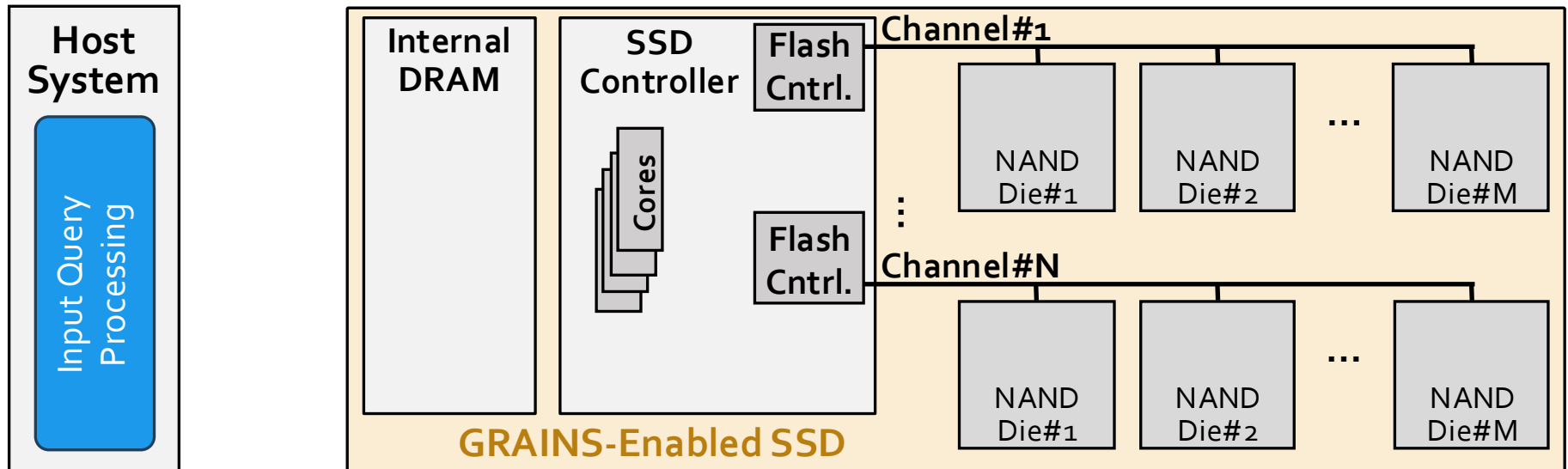
GRAINS

- The *first* system for analysis with large-scale genome graphs in storage
- Enabled via **storage-aware algorithm-architecture co-design**, based on the **properties of large-scale genome graphs** to
 - Make the execution pipeline **more storage-friendly**
 - Further improve **performance, energy-efficiency, and cost-effectiveness** via **storage-centric computing**



GRAINS's Algorithm-Architecture Co-Design

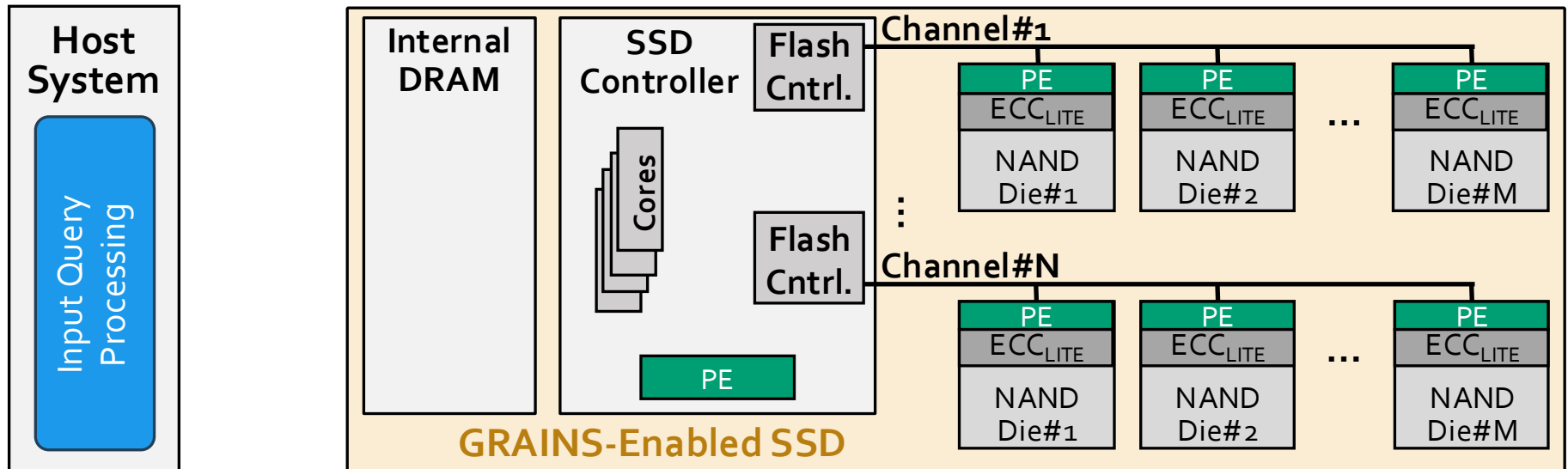
New batching technique and execution flow
*based on the unique features of genome graphs
that reduces #random accesses*



GRAINS's Algorithm-Architecture Co-Design

New batching technique and execution flow
*based on the unique features of genome graphs
that reduces #random accesses*

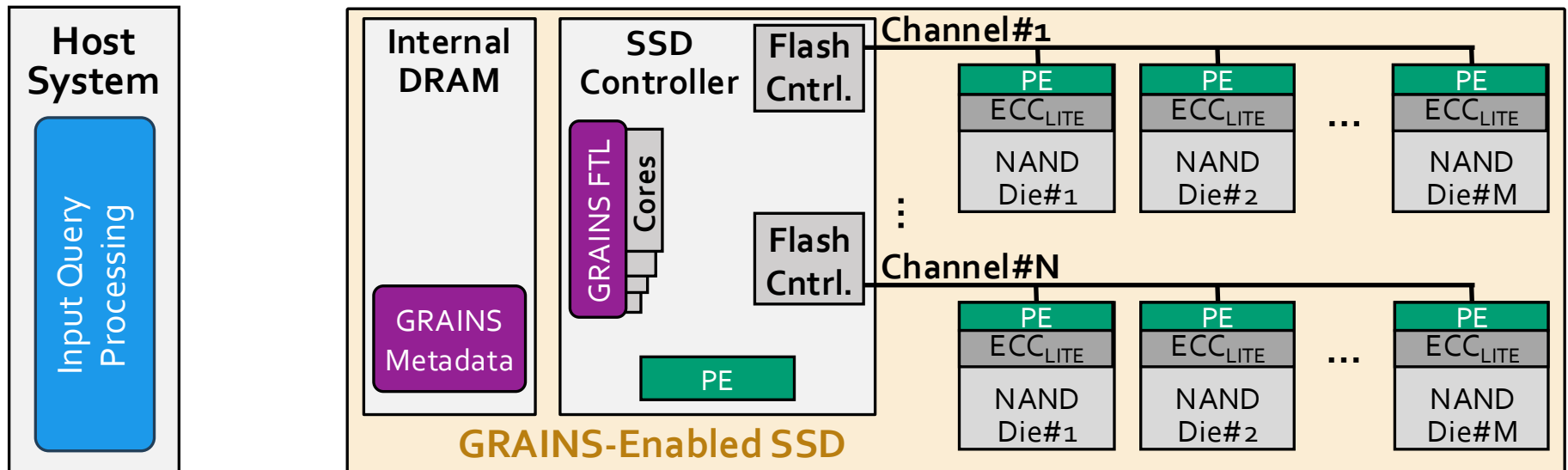
In-storage and In-flash processing
*to avoid transferring low-reused pages
and bandwidth waste*



GRAINS's Algorithm-Architecture Co-Design

New batching technique and execution flow
*based on the unique features of genome graphs
that reduces #random accesses*

In-storage and In-flash processing
*to avoid transferring low-reused pages
and bandwidth waste*



FTL and data placement
*based on genome graph properties,
to leverage SSD's full internal bandwidth*

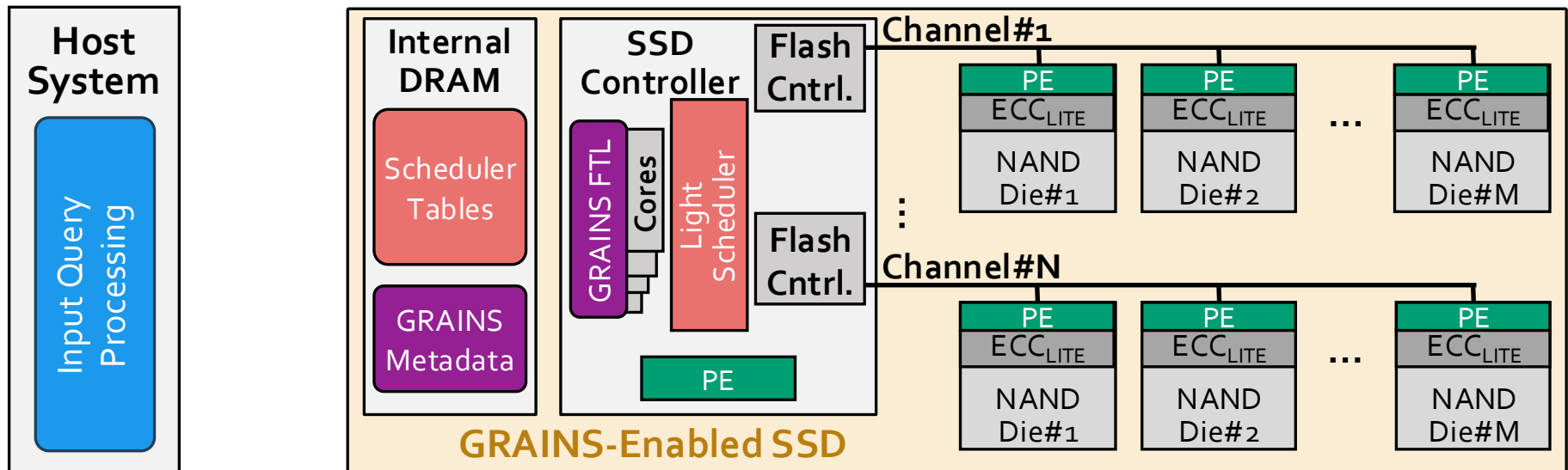
GRAINS's Algorithm-Architecture Co-Design

New batching technique and execution flow

*based on the unique features of genome graphs
that reduces #random accesses*

In-storage and In-flash processing

*to avoid transferring low-reused pages
and bandwidth waste*



FTL and data placement

*based on genome graph properties,
to leverage SSD's full internal bandwidth*

Lightweight scheduling technique

*enabled by repurposing the existing SSD structures
to leverage the full parallelism of all flash dies*

Evaluation Methodology Overview (I)

Performance, Energy, and Power Analysis

Hardware Components

- Synthesized Verilog model for the in-storage accelerators
- 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® host CPU with 128 physical cores
- 1.5 TB host DRAM

Baseline Comparison Points

- **Fulgor** [WABI'23]: A state-of-the-art node-centric tool
- **MetaGraph** [Nature'25]: A state-of-the-art edge-centric tool
- **AccIdealMem**: An idealized accelerator for graph queries with no main memory overheads, but still suffers from the I/O overhead of moving a large amount of low-reuse data

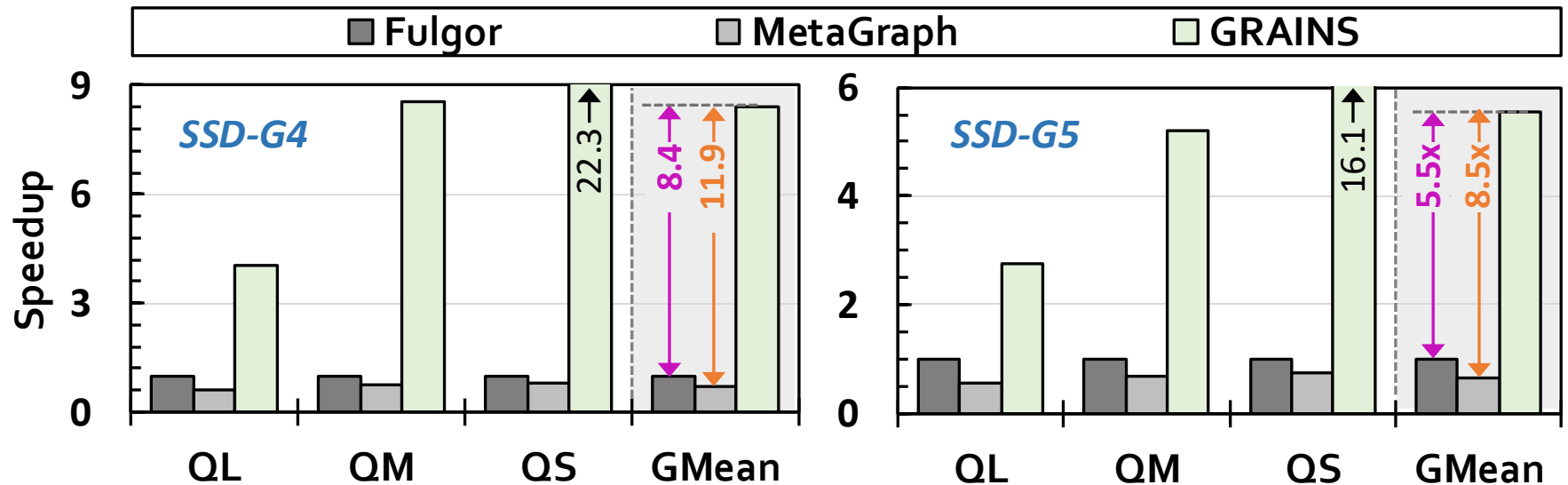
SSD Configurations

- **SSD-G4**: With PCIe Gen4 interface (7 GB/s sequential read bandwidth)
- **SSD-G5**: With PCIe Gen5 interface (14.8 GB/s sequential read bandwidth)

SAFARI



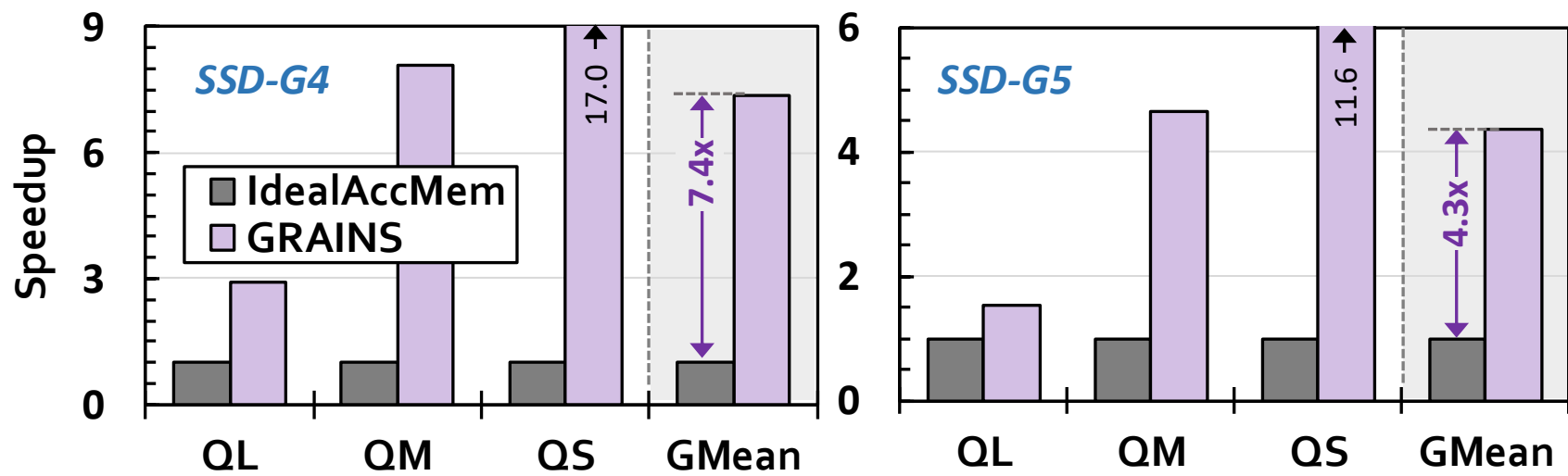
Benefits over Software



GRAINS provides significant speedup over **Fulgor** and **Metagraph** baselines, with both **SSD-G4** and **SSD-G5**

GRAINS improves average **energy-efficiency** by **11.3x** and **21.5x** over **Fulgor** and **Metagraph**, respectively

Benefits over Hardware



GRAINS provides significant speedups over
AccIdealMem baseline

GRAINS improves average **energy-efficiency**
by **3.1—20.7x** over **AccIdealMem**

Area and Power

- Based on **Synthesis** of **GRAINS's hardware units** using the Synopsys Design Compiler @ 22nm technology node
- ECC_{LITE} area based on the original paper [AiF, ISCA'25]

Logic unit	Area [mm ²]	Power [mW]
On SSD Controller	0.0025	0.21
On Die (ECC_LITE)	0.036	18.04
On Die (Others)	0.000093	0.01

On-controller logic units are 0.7% of the four cores on an SSD controller

On-die logic units take only 0.2% of the flash die's area
and fit well within its power budget

Our Approach

Designing customized *storage-centric systems* that efficiently

① *analyze* genomic and metagenomic data *inside the storage system*

① **GenStore** [ASPLOS'22]

In-storage filtering
for **genomic** analysis

② **MegIS** [ISCA'24]

Cooperative in-storage processing
for **metagenomic** analysis

③ **GRAINS** [ISCA'26]

Storage-aware algorithm-architecture co-design
for **graph-based** (meta)genomic analyses

② *enable highly-compressed storage and high-performance access
of large-scale sequence data*

④ **SAGe** [HPCA'26]

Algorithm-architecture co-design
for **storing and accessing** sequence data

Our Approach

Designing customized **storage-centric systems** that efficiently

① **analyze** genomic and metagenomic data **inside the storage system**

① **GenStore** [ASPLOS'22]

In-storage filtering
for **genomic** analysis

② **MegIS** [ISCA'24]

Cooperative in-storage processing
for **metagenomic** analysis

③ **GRAINS** [ISCA'26]

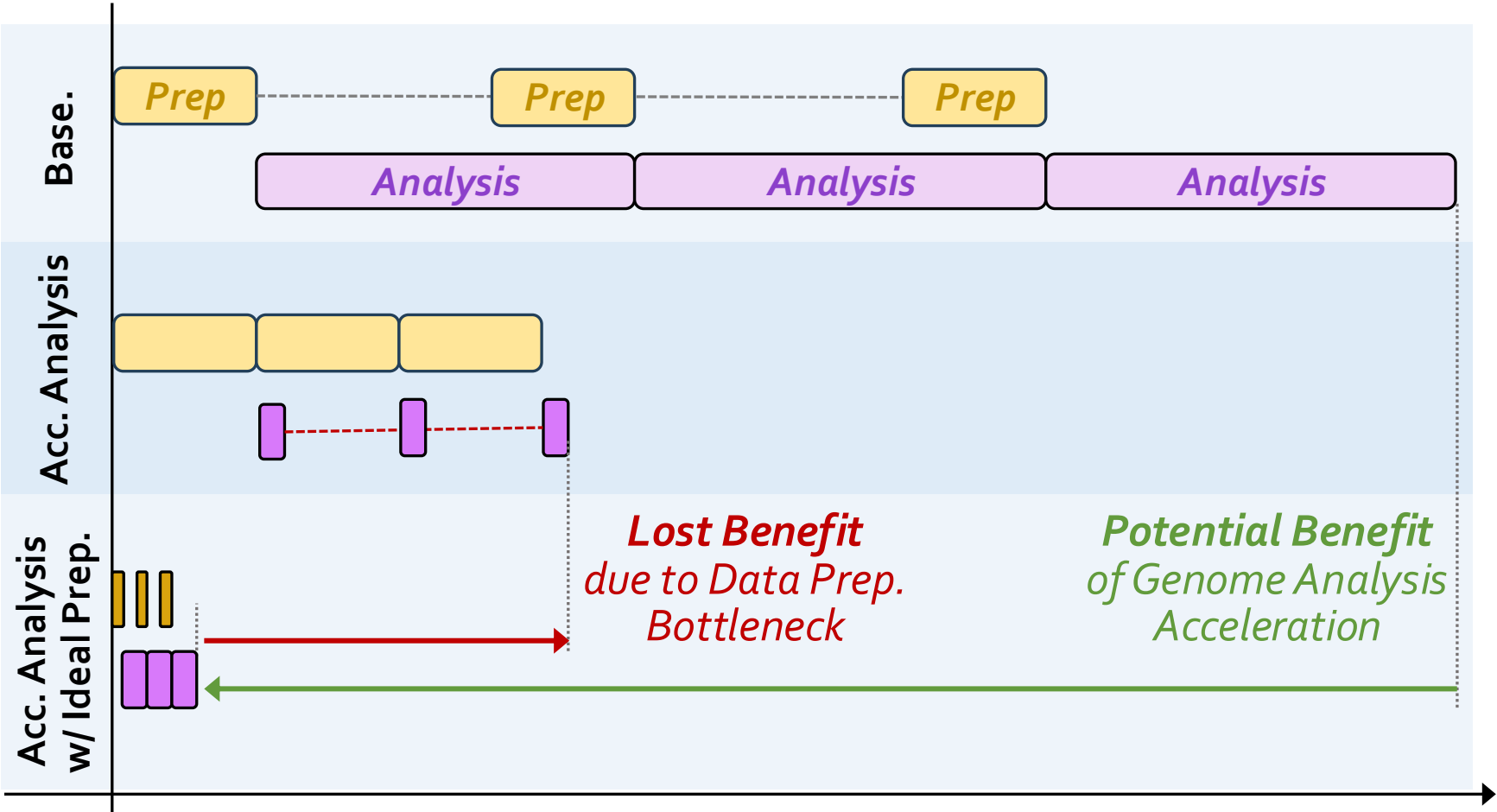
Storage-aware algorithm-architecture co-design
for **graph-based** (meta)genomic analyses

② **enable highly-compressed storage and high-performance access**
of large-scale sequence data

④ **SAGe** [HPCA'26]

Algorithm-architecture co-design
for **storing** and **accessing** sequence data

Recall: Effect of Data Preparation Bottleneck



Execution Timeline
(Not drawn to scale)

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

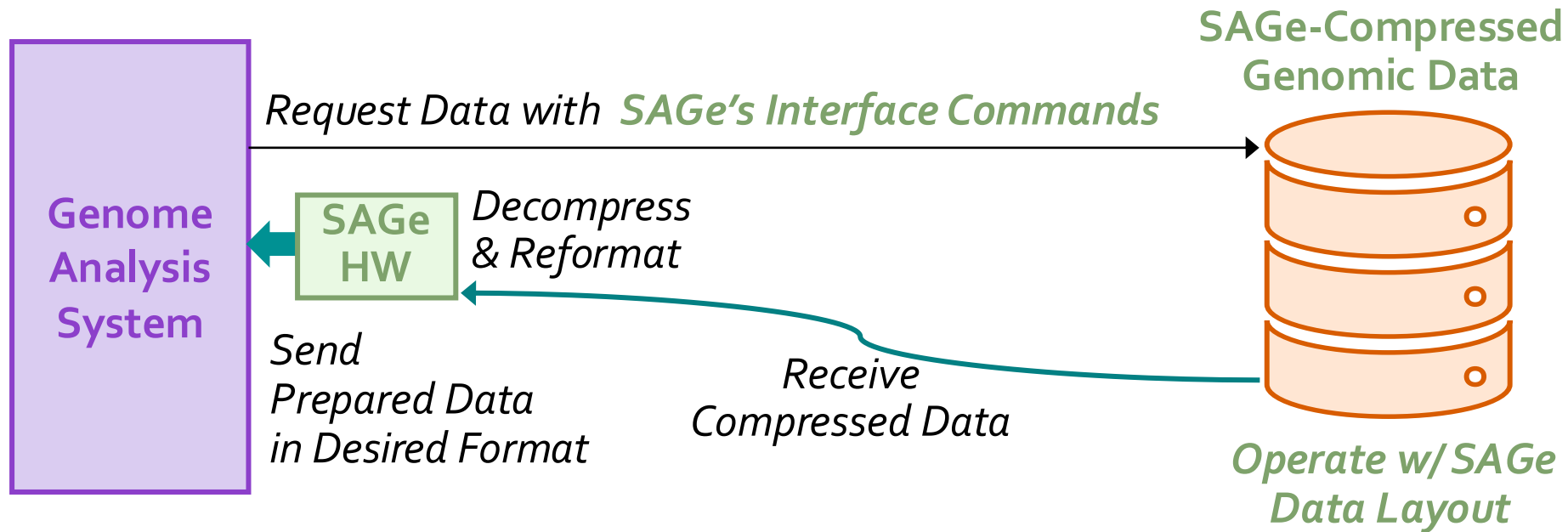
Meeting all these requirements is challenging

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



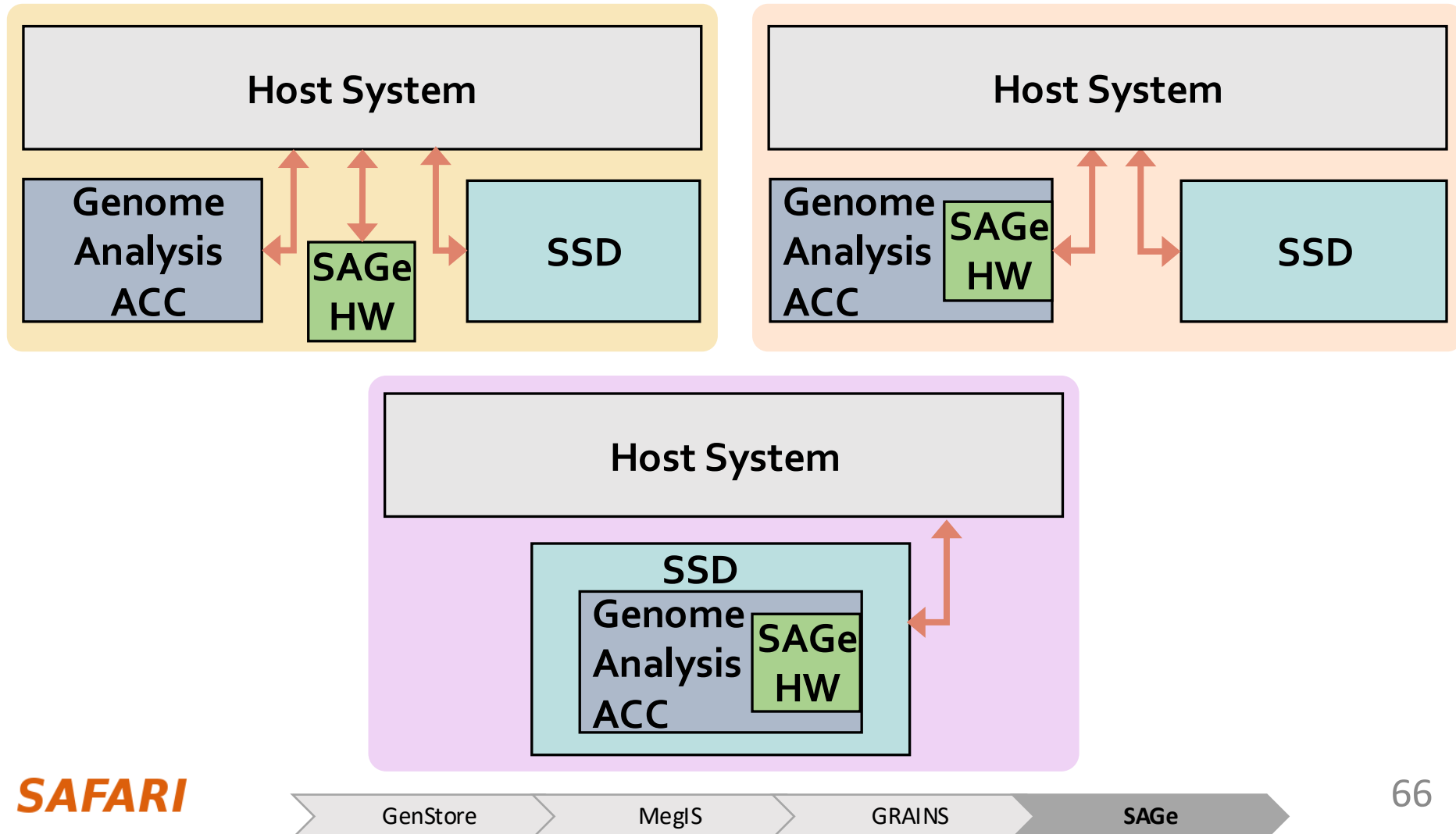
An algorithm-architecture co-design for *highly-compressed storage* and *high-performance access* of large-scale *genomic sequence data*

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



Case Studies of SAGe's Hardware Integration

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



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® host CPU with 128 physical cores
- 1.5 TB host DRAM

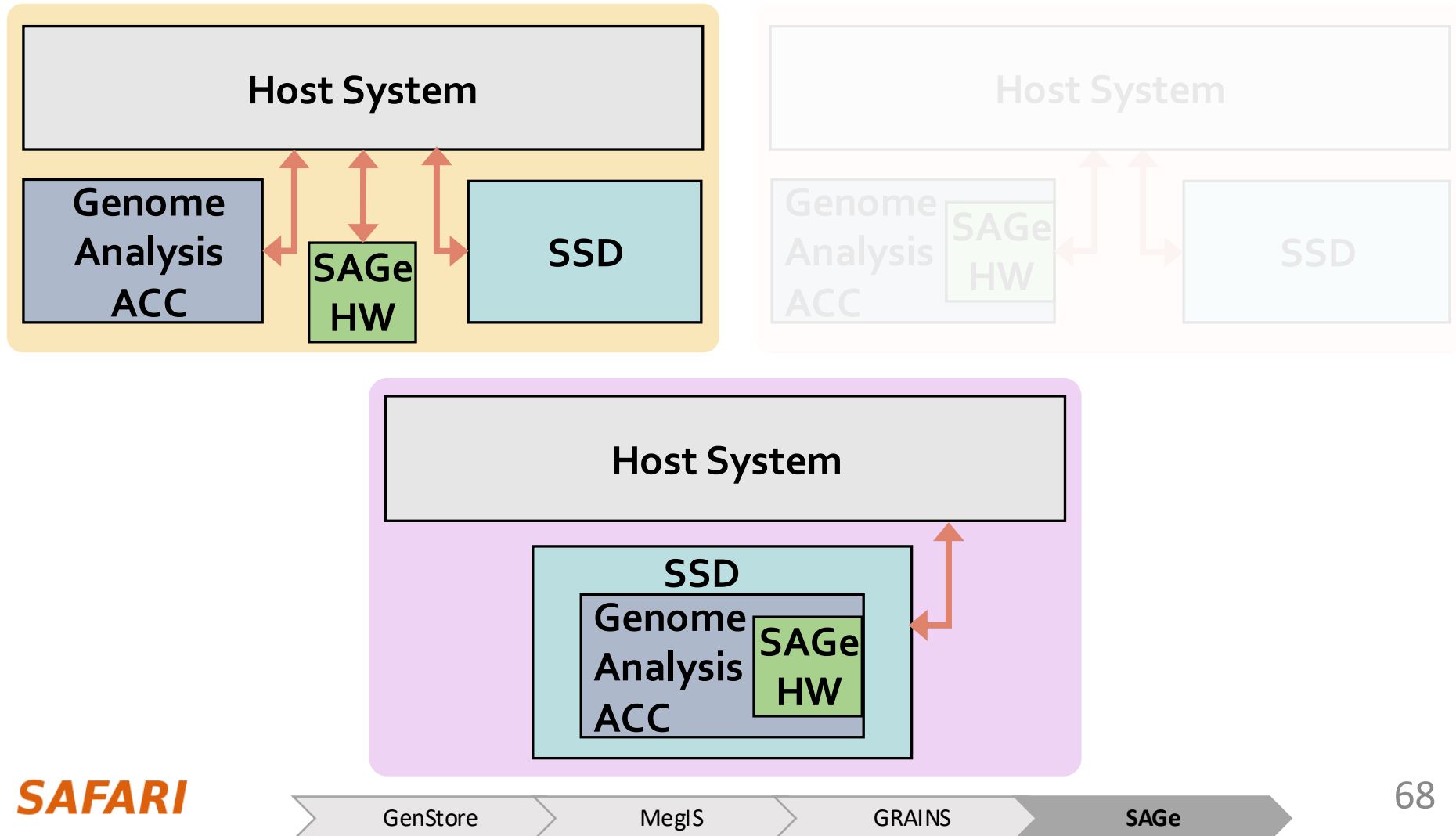
Datasets: Read sets (RS)

- 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

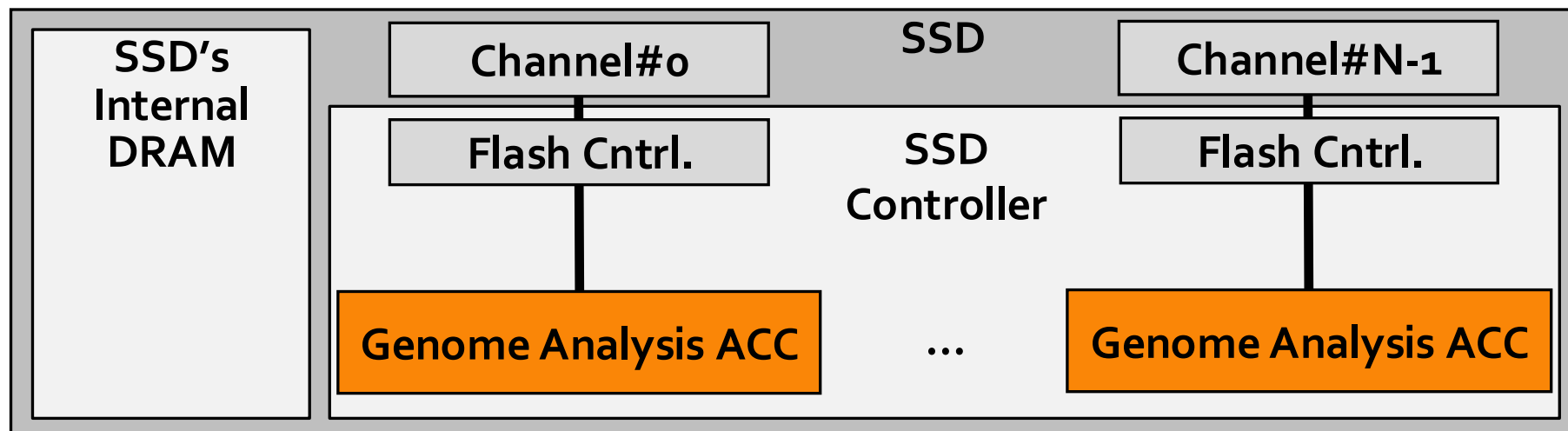
Due to its lightweight design, SAGe can seamlessly integrate with genome analysis systems



Evaluation Methodology Overview (II)

Genome Analysis

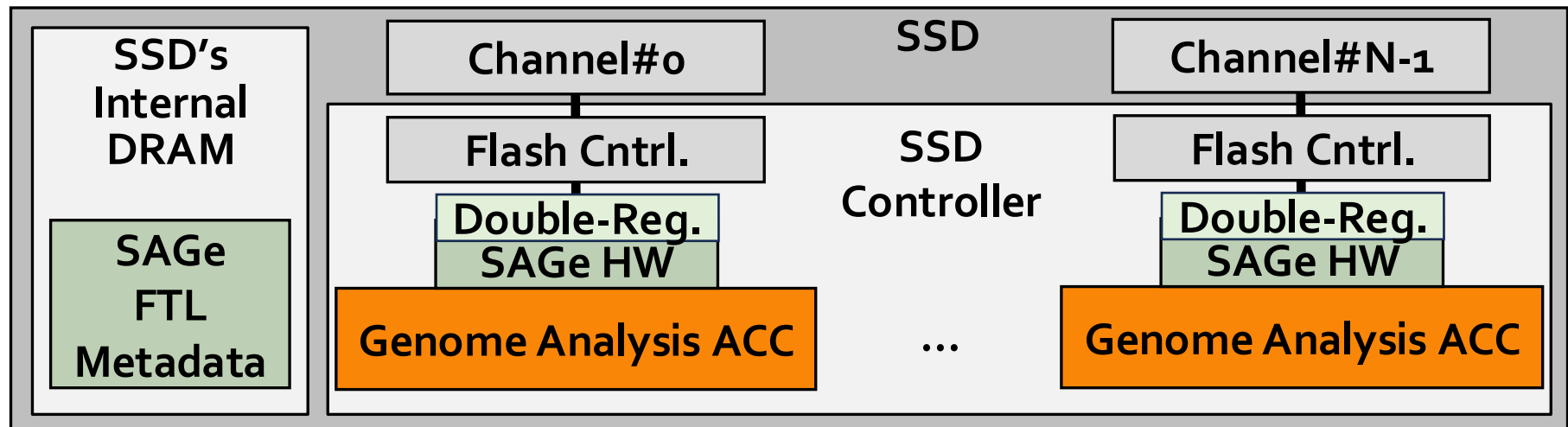
- A state-of-the-art hardware accelerator (GEM [Chen+, TPDS'23]), for read mapping
- Near-data processing (NDP) baseline: GenStore [Mansouri Ghiasi+, ASPLOS'22]
 - Alleviates the burden of moving and analyzing a large amount of low-reuse data
 - **Implemented in the resource-constrained environment of the SSD**



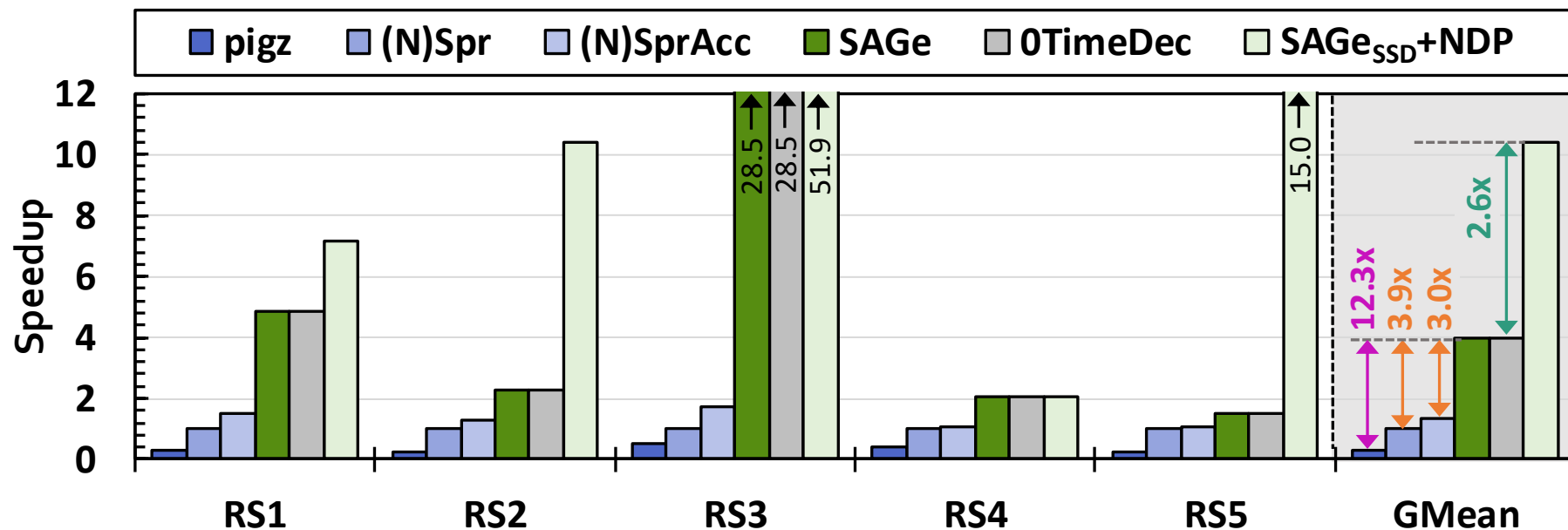
Evaluation Methodology Overview (II)

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 a standalone accelerator for data preparation
- **SAGe_{SSD}**: SAGe inside the SSD, for integration with the NDP analysis system



Speedup



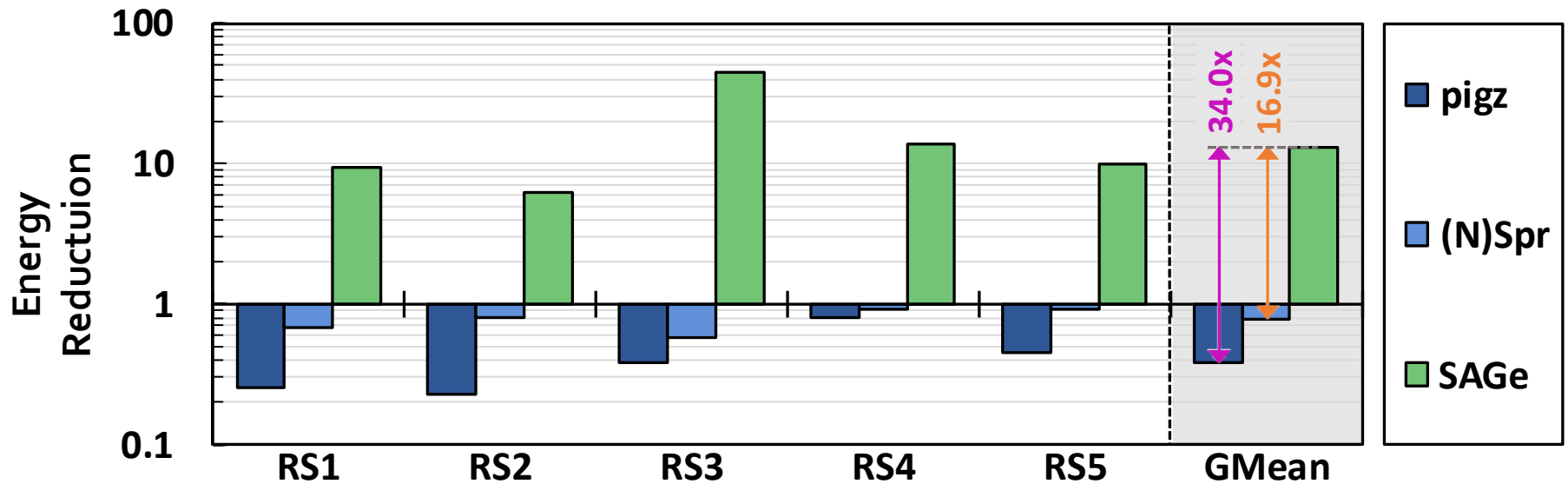
SAGe provides significant speedup over both **general-purpose** and **genomics-specific** baselines

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

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 [mm ²]	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	0.000020	0.035
Control Unit	1 per channel	0.000029	0.025
<i>Total for an 8-channel SSD</i>	-	0.002	0.77

SAGe's hardware units consume small area and power

Our Approach

Designing customized *storage-centric systems* that efficiently

① *analyze* genomic and metagenomic data *inside the storage system*

① **GenStore** [ASPLOS'22]

In-storage filtering
for **genomic** analysis

② **MegIS** [ISCA'24]

Cooperative in-storage processing
for **metagenomic** analysis

③ **GRAINS** [ISCA'26]

Storage-aware algorithm-architecture co-design
for **graph-based** (meta)genomic analyses

② *enable highly-compressed storage and high-performance access
of large-scale sequence data*

④ **SAGe** [HPCA'26]

Algorithm-architecture co-design
for **storing** and **accessing** sequence data

Conclusion

By designing customized **storage-centric systems** that efficiently

① **analyze** genomic and metagenomic data **inside the storage system**

① **GenStore**

② **MegIS**

③ **GRAINS**

② enable **highly-compressed storage** and **high-performance access** of large-scale sequence data

④ **SAGe**



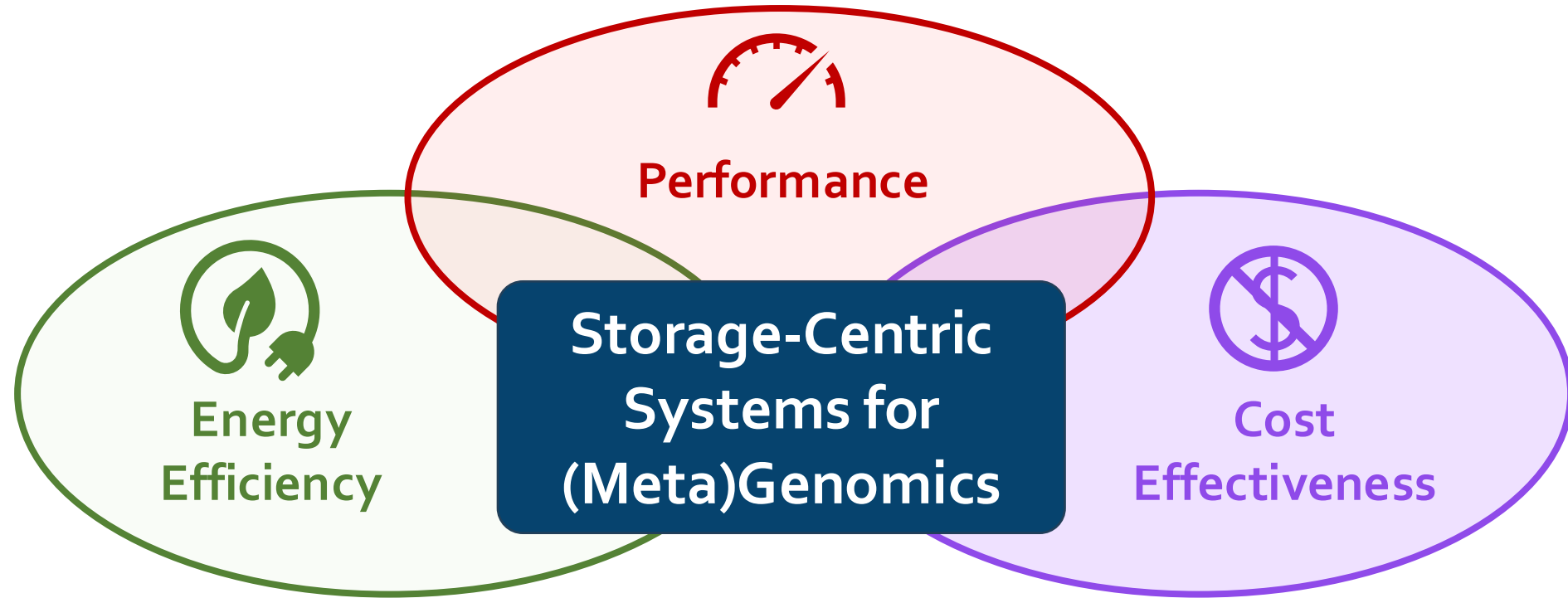
we can alleviate the overheads of

data movement

computation

data preparation

Putting It All Together



*Make accurate (meta)genomic analysis
more accessible for wider adoption*

More About My Research



<https://sites.google.com/view/nikamansourighiasi>

- Thesis Publications
- Other Storage-Centric Computing Works
- Memory-Centric Computing & Dense-3D Integration
- Bioinformatics Algorithms
- Algorithm-Architecture Co-design
- Improving Storage & Memory System
- Architecture for Health (Arch4Health) Initiative

Storage-Centric System Designs for Enabling Fast, Efficient, and Low-Cost Genomic and Metagenomic Analyses

Nika Mansouri Ghiasi

Onur Mutlu

Arch4Health @ ICS 2026
June 2026

SAFARI

ETH zürich