AMERICAN UNIVERSITY OF BEIRUT

FACULTY OF ARTS & SCIENCES

ETH zürich

HIGH-THROUGHPUT SEQUENCE ALIGNMENT USING REAL PROCESSING-IN-MEMORY SYSTEMS

Safaa Diab, Amir Nassereldine, Mohammed Alser, Juan Gómez Luna, Onur Mutlu, Izzat El Hajj

June 18, 2023

Real-world Processing-in-Memory Systems for Modern Workloads (at ISCA'23)

AMERICAN UNIVERSITY OF BEIRUT
FACULTY OF ARTS & SCIENCES

ETH zürich

High-throughput Sequence Alignment using Real Processing-in-Memory Systems

- Published in Bioinformatics (2023)

A Framework for High-throughput Sequence Alignment using Real Processing-in-Memory Systems

Safaa Diab [1] Amir Nassereldine [1] Mohammed Alser [2] Juan Gómez Luna [2]
Onur Mutlu [2] Izzat El Hajj [1]

[1]American University of Beirut [2]ETH Zürich

https://arxiv.org/pdf/2208.01243.pdf
Source code: https://github.com/safaad/aim

Summary

- Sequence alignment is **memory-bound** on traditional processor-centric systems
- **Processing-in-memory (PIM)** overcomes the memory bandwidth bottleneck by placing cores near the memory
- We present **Alignment-in-Memory (AIM)**, a framework for sequence alignment on real PIM systems
  - Supports multiple alignment algorithms: NW, SWG, GenASM, and WFA
  - Implemented on UPMEM, the first real PIM system
- Results show **substantial speedups over CPUs and GPUs**

Outline

- Background on sequence alignment
- Processing-in-memory
- Our framework
- Evaluation

Outline

- **Background on sequence alignment**
- Processing-in-memory
- Our framework
- Evaluation

Genome Analysis and Sequence Alignment

- A **genome** is the complete set of an organism's genetic instructions
  - DNA is a string of base pairs (characters): A, C, G, T
  - The human genome contains 3.2 billion base pairs

- **Genome analysis** helps to understand genetic variations, predict the presence and abundance of microbes, monitor disease outbreaks, develop personalized medicine, etc.

- **Sequencing** machines provide only randomized fragments (**reads**) of the genome
  - Depending on the sequencing technology, there are short (50-300 bp) and long reads (10K-100K bp)
  - Need to be mapped to the reference genome

- A key step in the read mapping process is **sequence alignment**
  - Quadratic-time dynamic programming algorithms, e.g., Smith-Waterman, Needleman-Wunsch
  - State-of-the-art algorithms: GenASM, WFA, WFA-adaptive

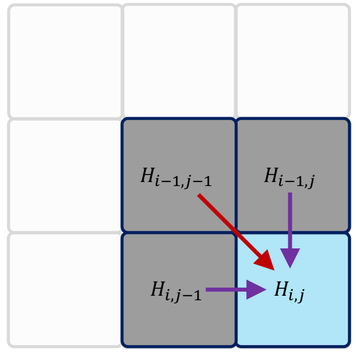
Sequence Alignment Example with the Smith-Waterman Algorithm

		A	A	T	C
	0	0	0	0	0
A	0				
A	0				
C	0				
C	0				

Align two sequences by filling a matrix that scores the similarity between the two sequences and their subsequences

Sequence Alignment Example with the Smith-Waterman Algorithm

		A	A	T	C
	0	0	0	0	0
A	0	3	3	1	0
A	0	3	6	4	2
C	0	1	4	3	7
C	0	0	2	1	6



Assign different scores for matches (+3), mismatches (-3), new gaps (-2), and continuing gaps (-1)

Take the maximum score for reaching a cell from previous cells

Sequence Alignment Example with the Smith-Waterman Algorithm

		A	A	T	C
	0	0	0	0	0
A	0	3	3	1	0
A	0	3	6	4	2
C	0	1	4	3	7
C	0	0	2	1	6

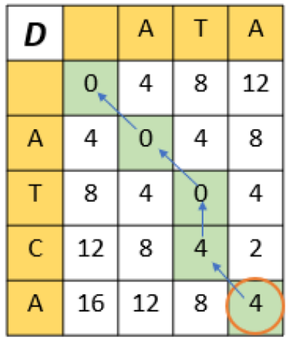
```
A A T C
| | | |
A A - C
```

Perform a traceback through the matrix to find the best alignment

AMERICAN UNIVERSITY OF BEIRUT
FACULTY OF ARTS & SCIENCES

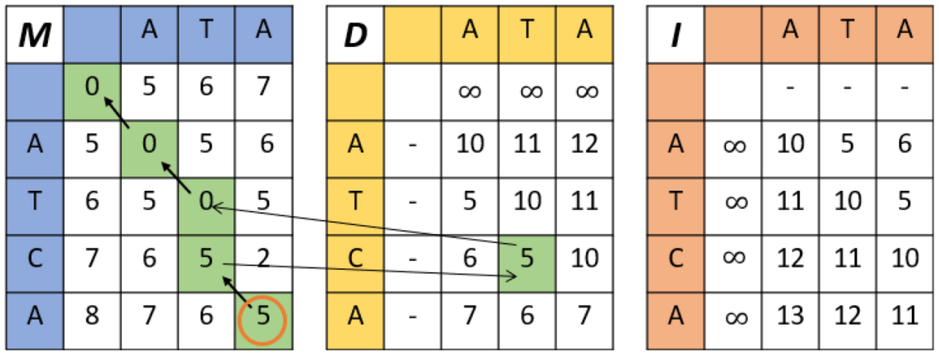
ETH zürich

Sequence Alignment Algorithms



D		A	T	A
	0	4	8	12
A	4	0	4	8
T	8	4	0	4
C	12	8	4	2
A	16	12	8	4

Needleman-Wunsch (NW)

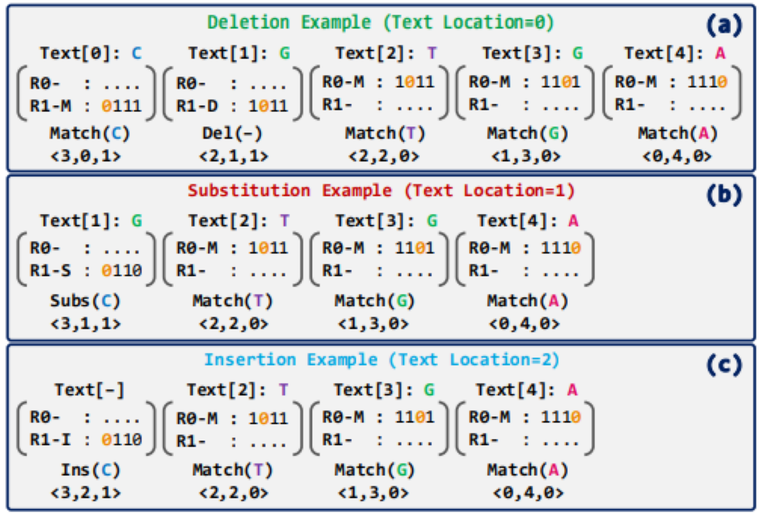


M		A	T	A
	0	5	6	7
A	5	0	5	6
T	6	5	0	5
C	7	6	5	2
A	8	7	6	5

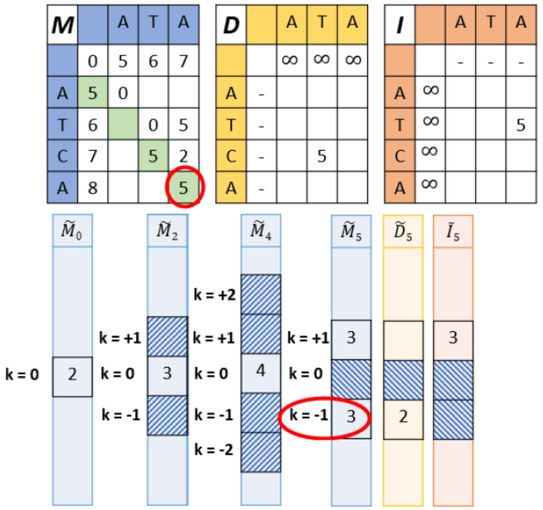
D		A	T	A
		∞	∞	∞
A	-	10	11	12
T	-	5	10	11
C	-	6	5	10
A	-	7	6	7

I		A	T	A
		-	-	-
A	∞	10	5	6
T	∞	11	10	5
C	∞	12	11	10
A	∞	13	12	11

Smith-Waterman-Gotoh (SWG)



GenASM

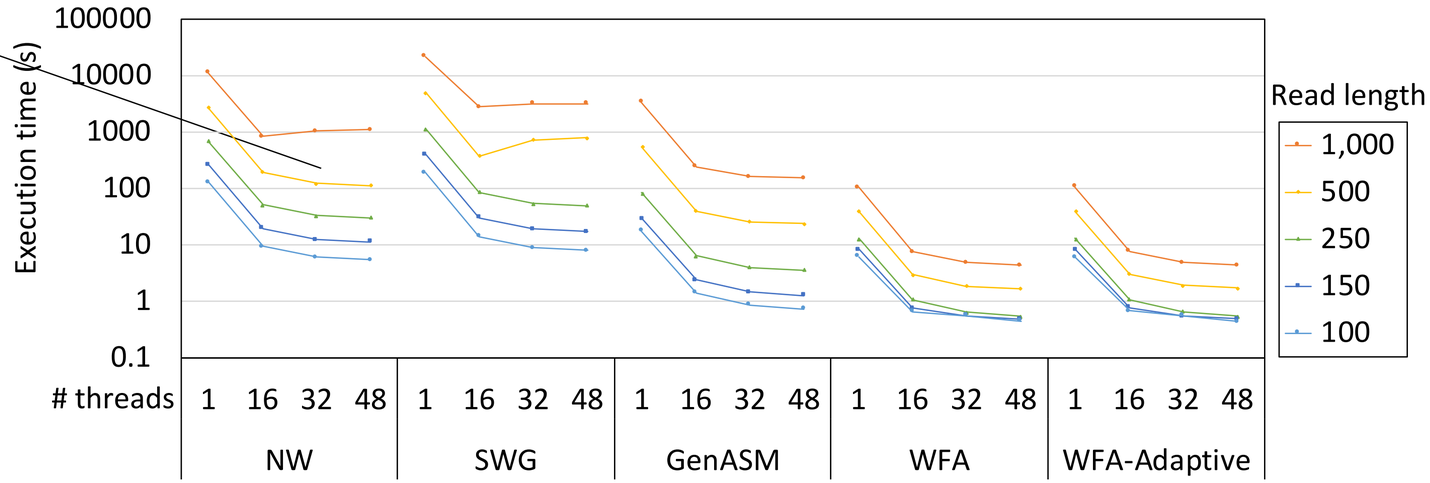


Wavefront Algorithm (WFA)

Cali, Damla Senol, et al. "GenASM: A high-performance, low-power approximate string matching acceleration framework for genome sequence analysis." *MICRO*, 2020.
Marco-Sola, Santiago, et al. "Fast gap-affine pairwise alignment using the wavefront algorithm." *Bioinformatics* 37.4 (2021): 456-463.

Sequence Alignment Scaling on CPUs

Observation: limited performance improvement as the number of CPU threads grows

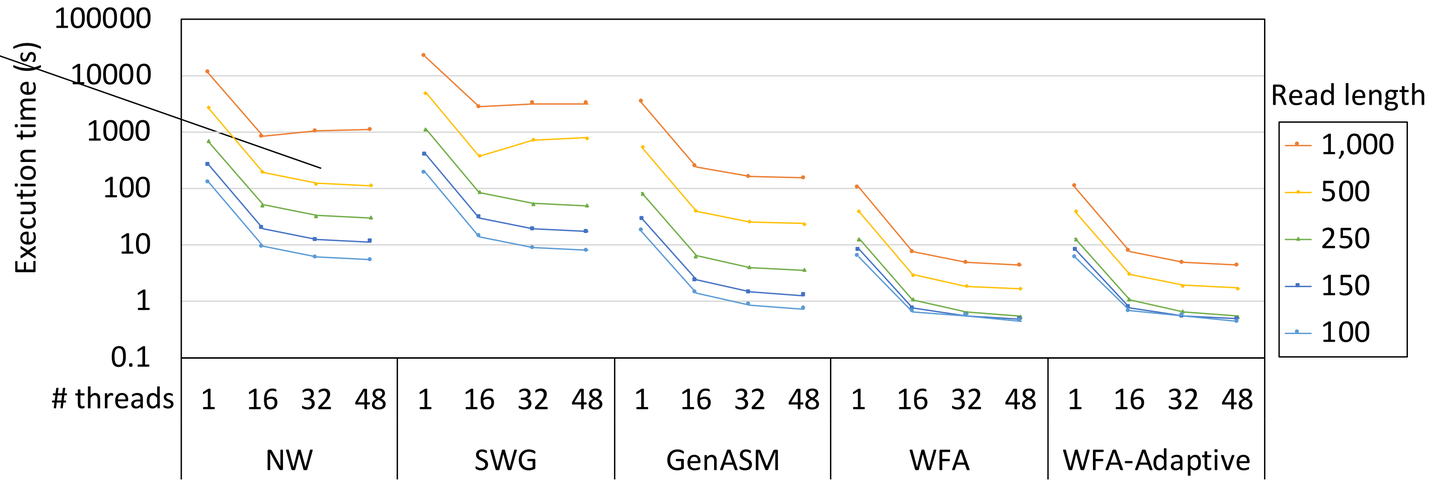


AMERICAN UNIVERSITY OF BEIRUT
FACULTY OF ARTS & SCIENCES

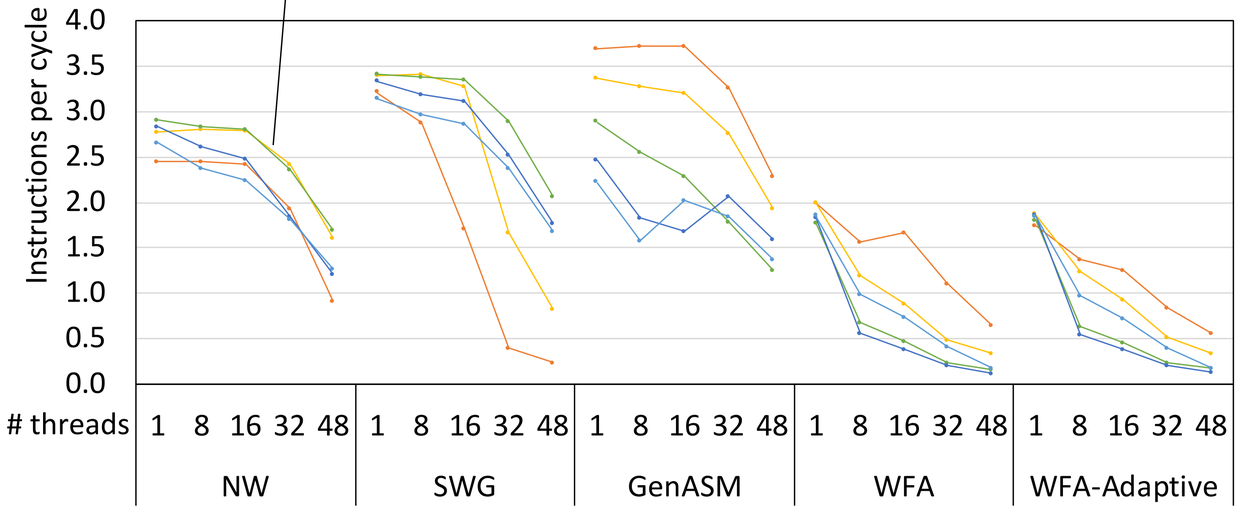
ETH zürich

Sequence Alignment Scaling on CPUs

Observation: limited performance improvement as the number of CPU threads grows



As the number of CPU threads grows, IPC decreases meaning threads spend **more time idle**

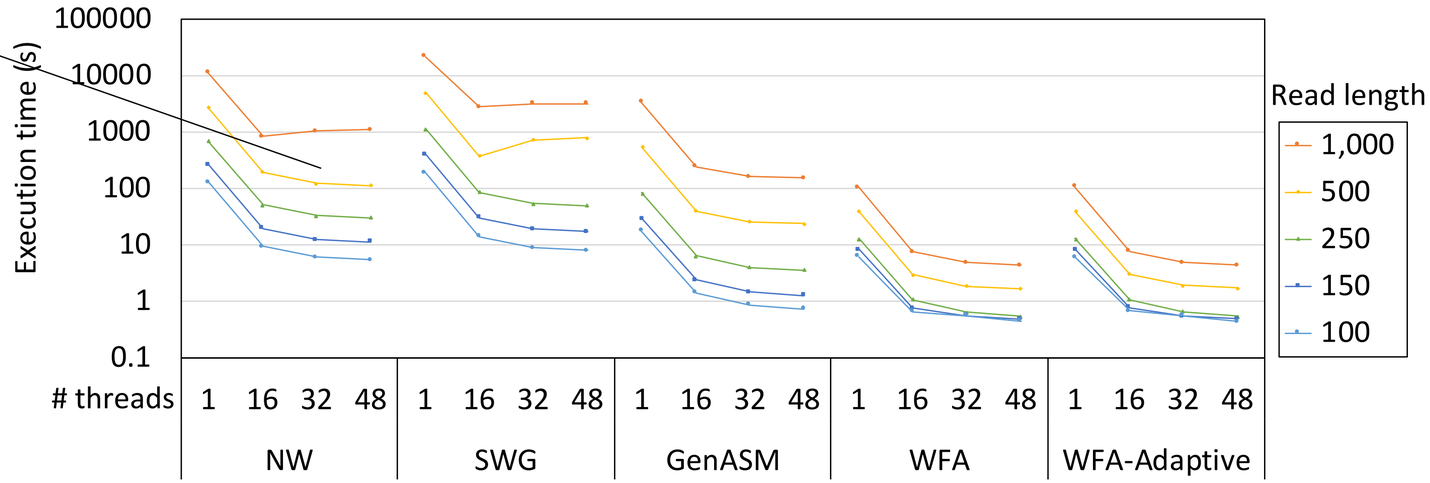


AMERICAN UNIVERSITY OF BEIRUT
FACULTY OF ARTS & SCIENCES

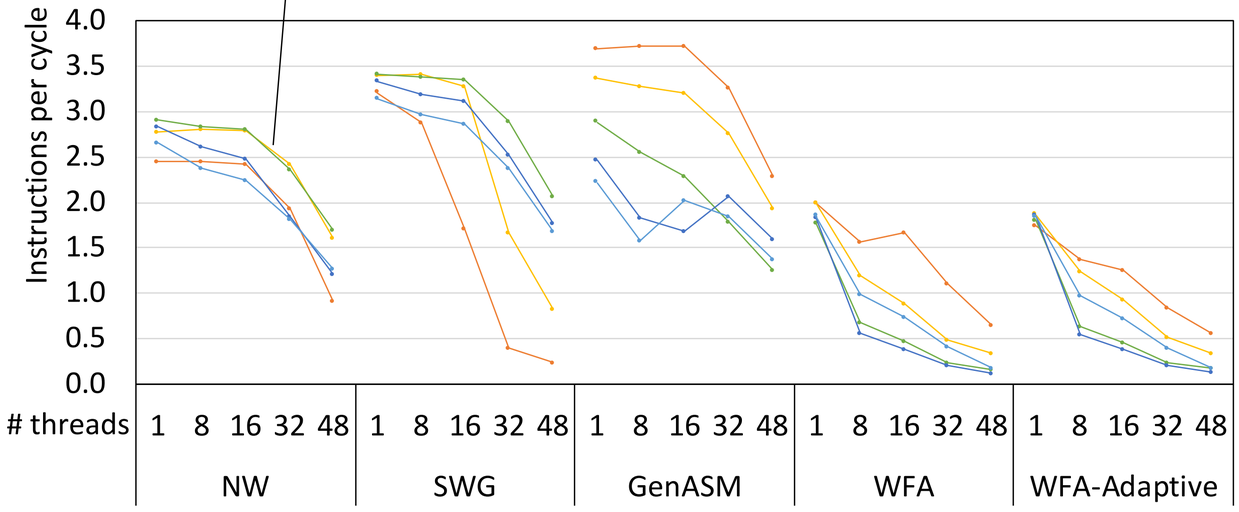
ETH zürich

Sequence Alignment Scaling on CPUs

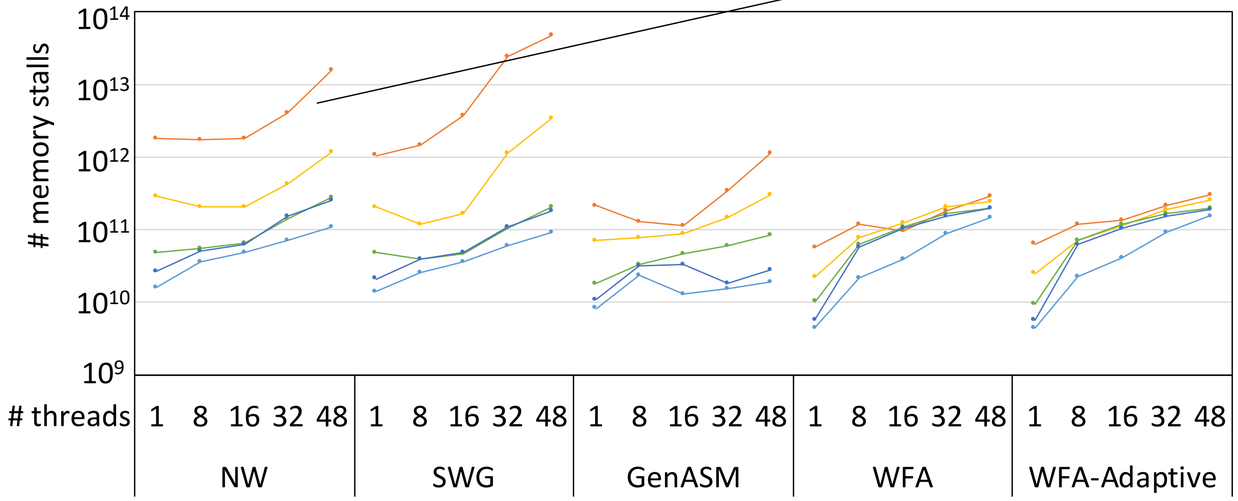
Observation: limited performance improvement as the number of CPU threads grows



As the number of CPU threads grows, IPC decreases meaning threads spend **more time idle**



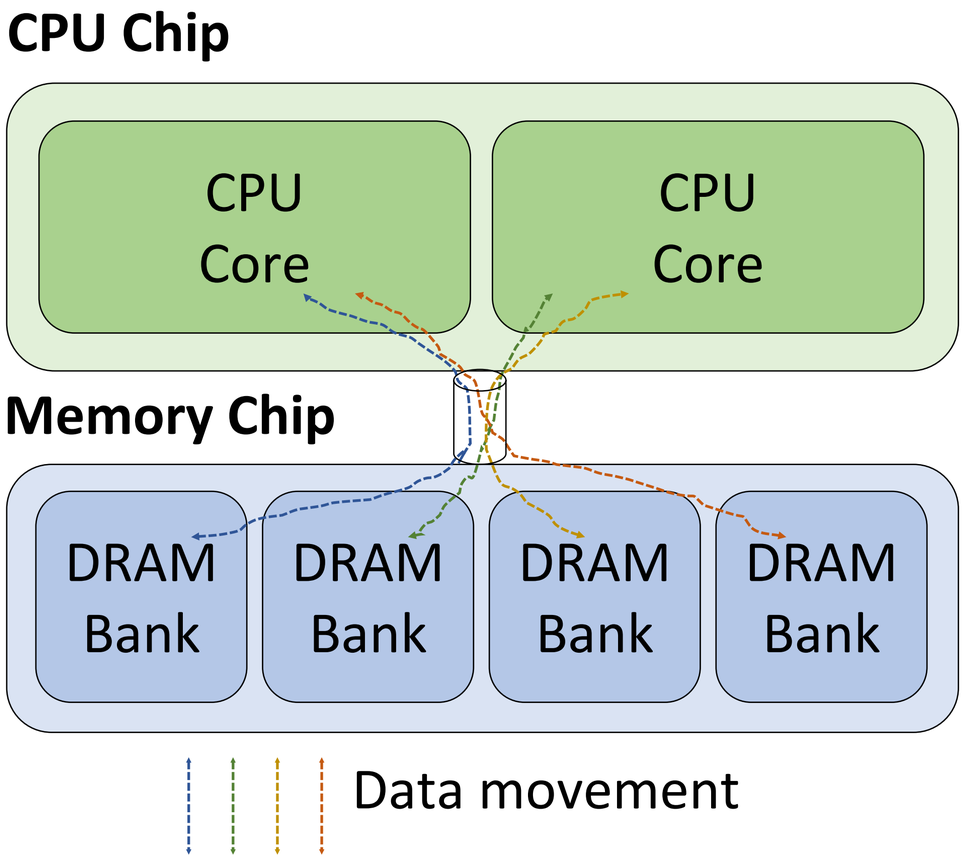
As the number of CPU threads grows, threads spend more time stalling **waiting for memory**



Outline

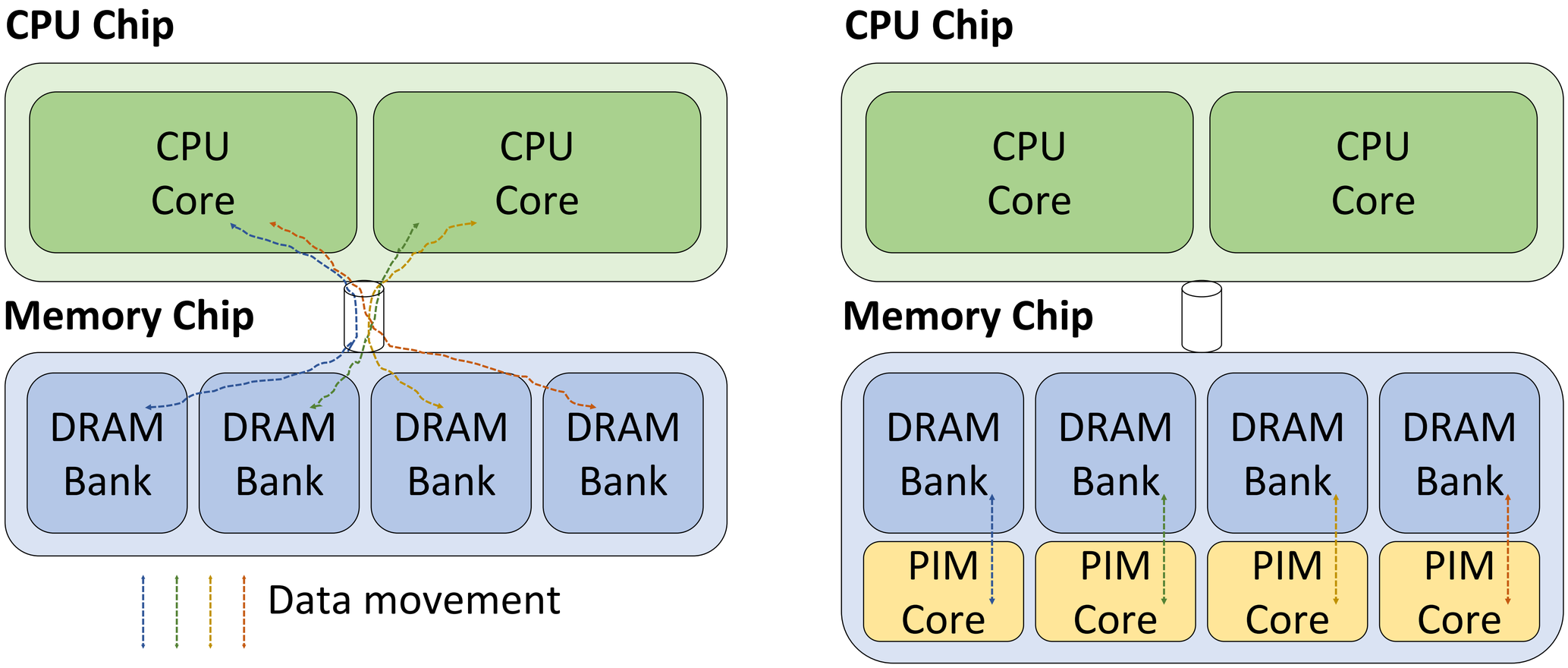
- Background on sequence alignment
- **Processing-in-memory**
- Our framework
- Evaluation

Memory Bandwidth Bottleneck



Conventional CPU processing

Processing-in-Memory



Conventional CPU processing

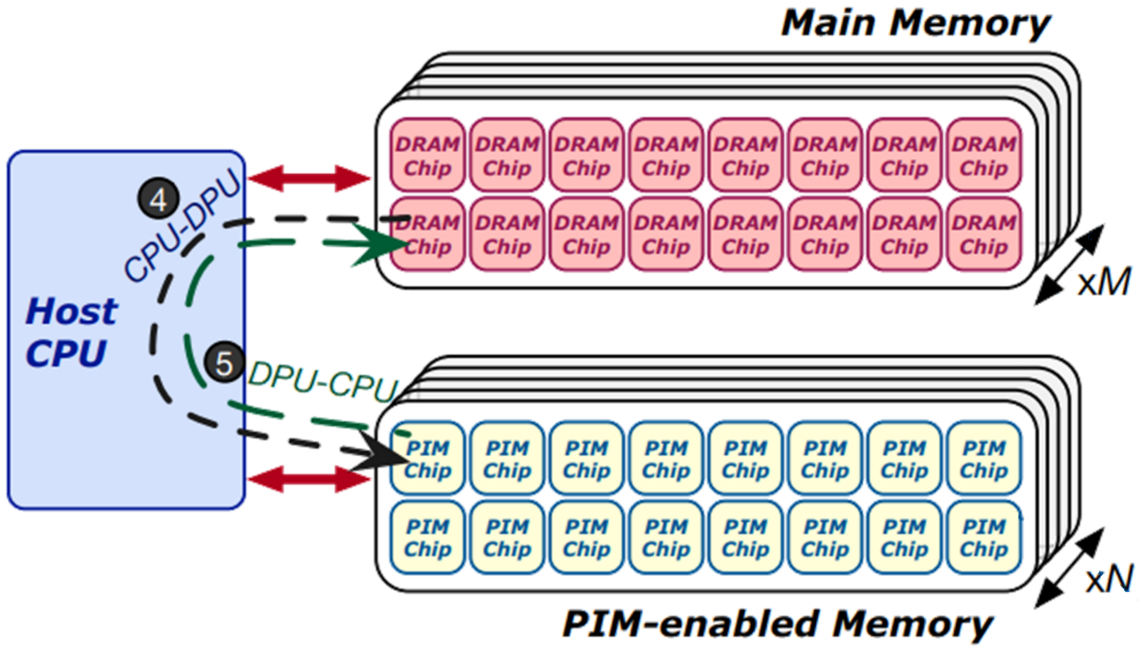
Processing-in-memory (PIM)

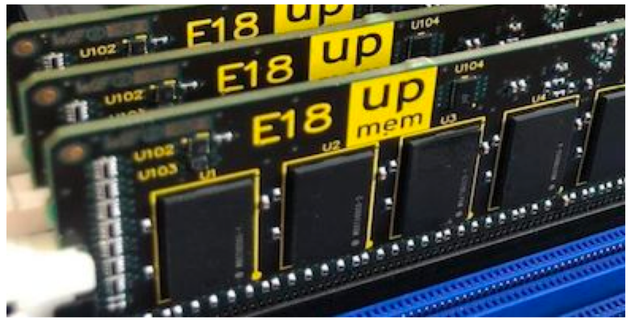
Real Processing-in-Memory Systems

- **Real-world PIM architectures** are becoming a reality
  - UPMEM PIM, Samsung HBM-PIM, Samsung AxDIMM, SK Hynix AiM, Alibaba HB-PNM

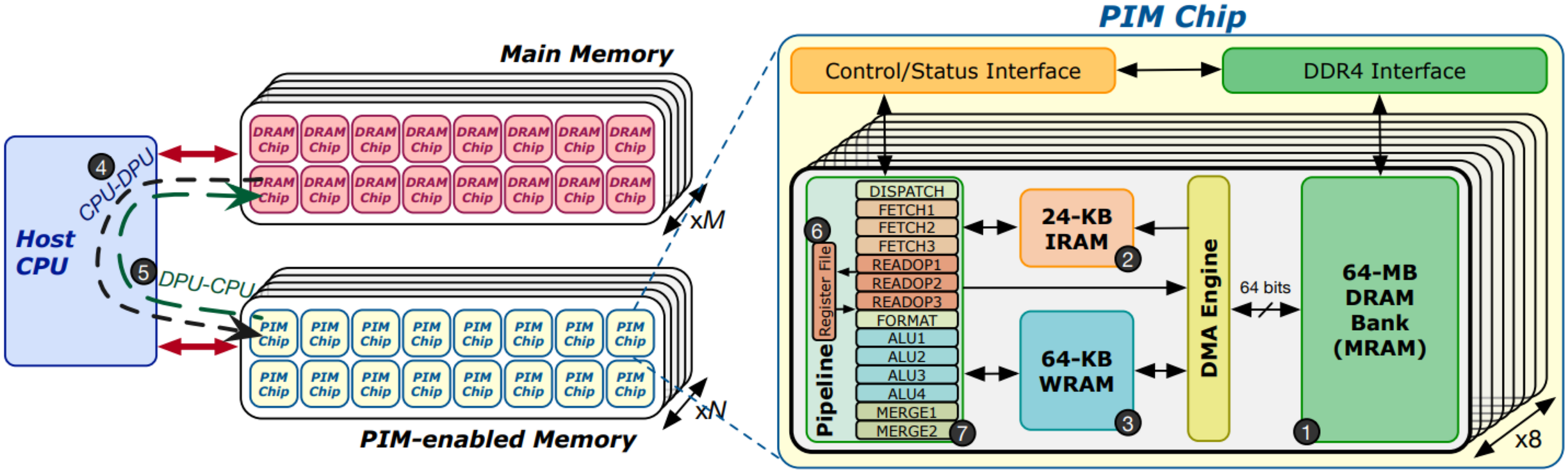
- These PIM systems have some common characteristics:
  1. There is a **host processor** (CPU or GPU) with access to (1) standard main memory, and (2) PIM-enabled memory
  2. PIM-enabled memory contains **multiple PIM processing elements** (PEs) with high bandwidth and low latency memory access
  3. PIM PEs run only at a few hundred MHz and have a **small number of registers and small (or no) cache/scratchpad**
  4. PIM PEs may need to **communicate via the host processor**

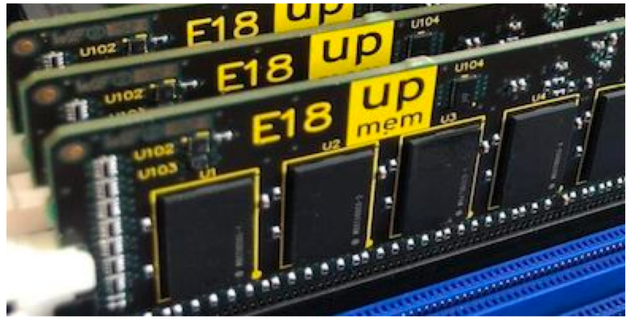
UPMEM: The First Real PIM Hardware





UPMEM: The First Real PIM Hardware

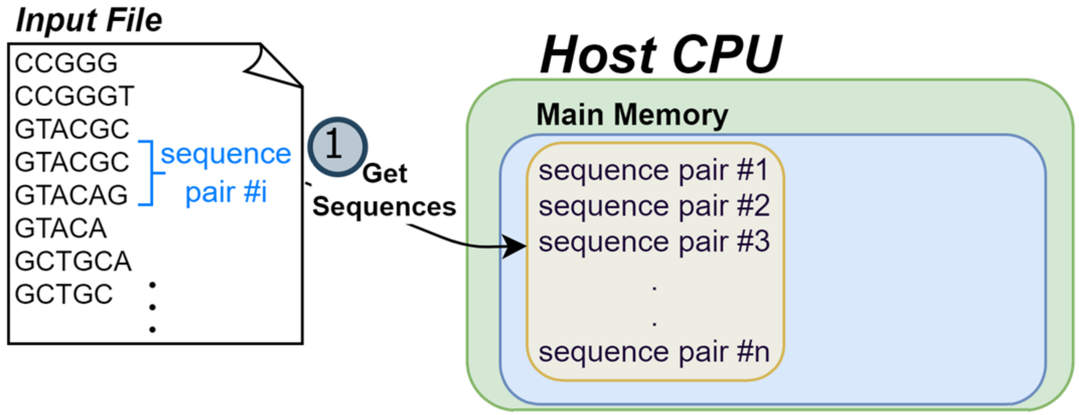




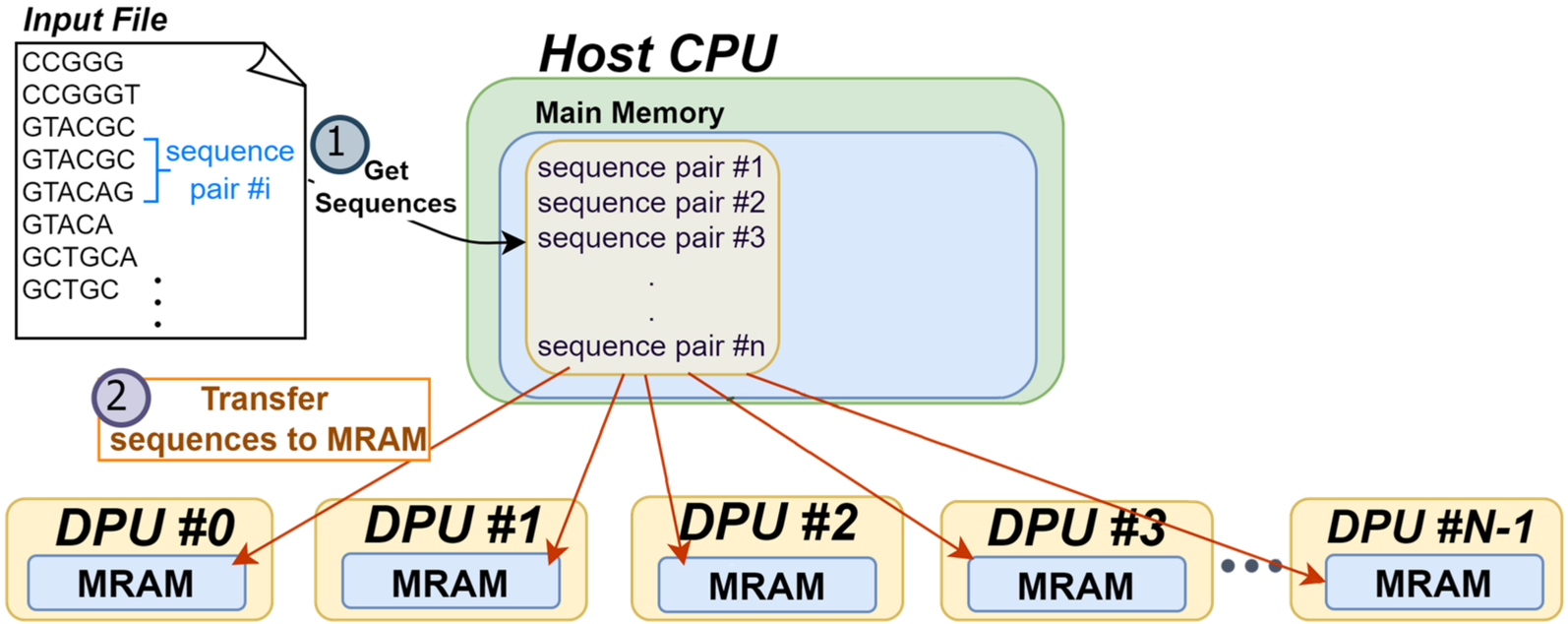
Outline

- Background on sequence alignment
- Processing-in-memory
- **Our framework**
- Evaluation

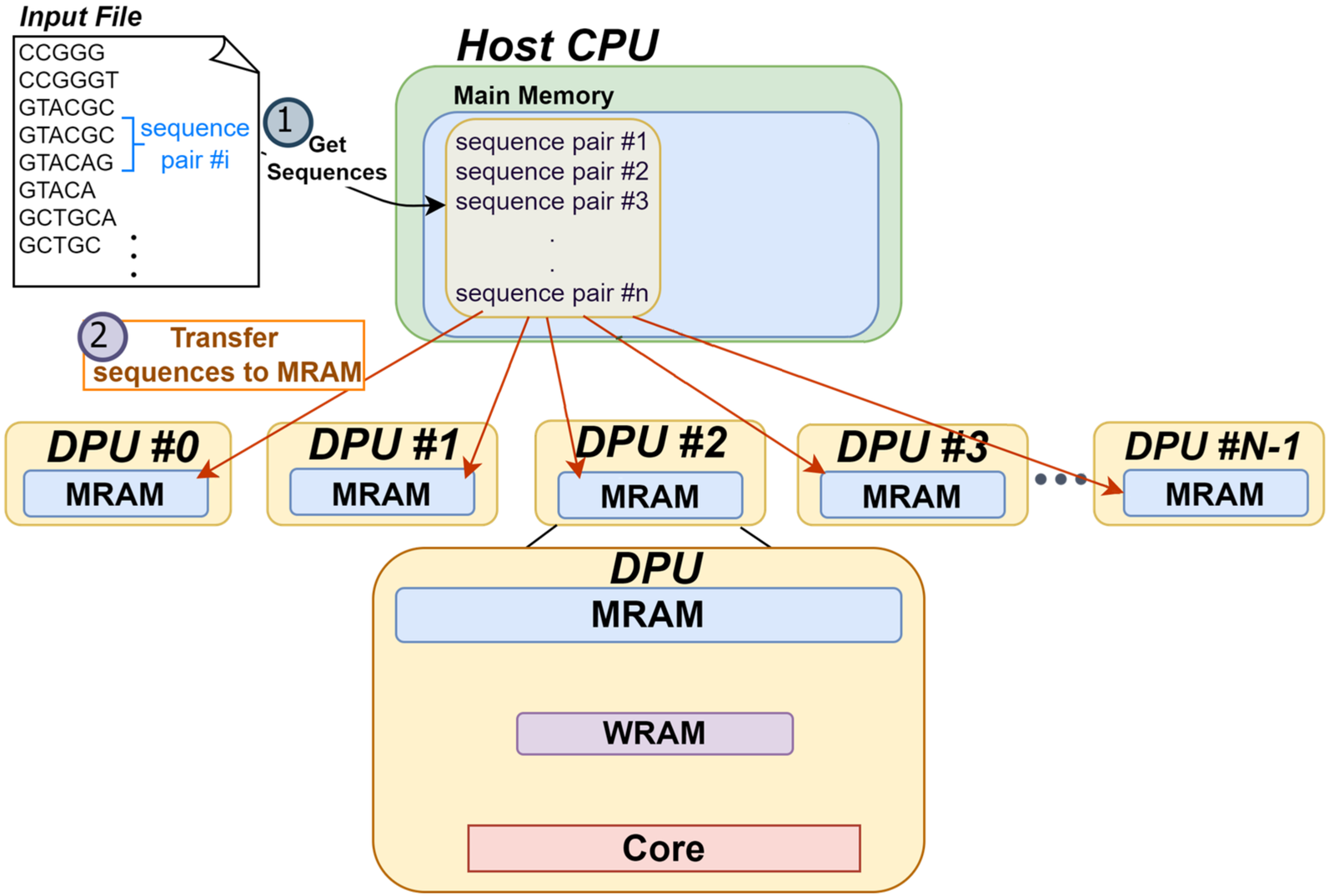
AIM: Alignment-in-Memory



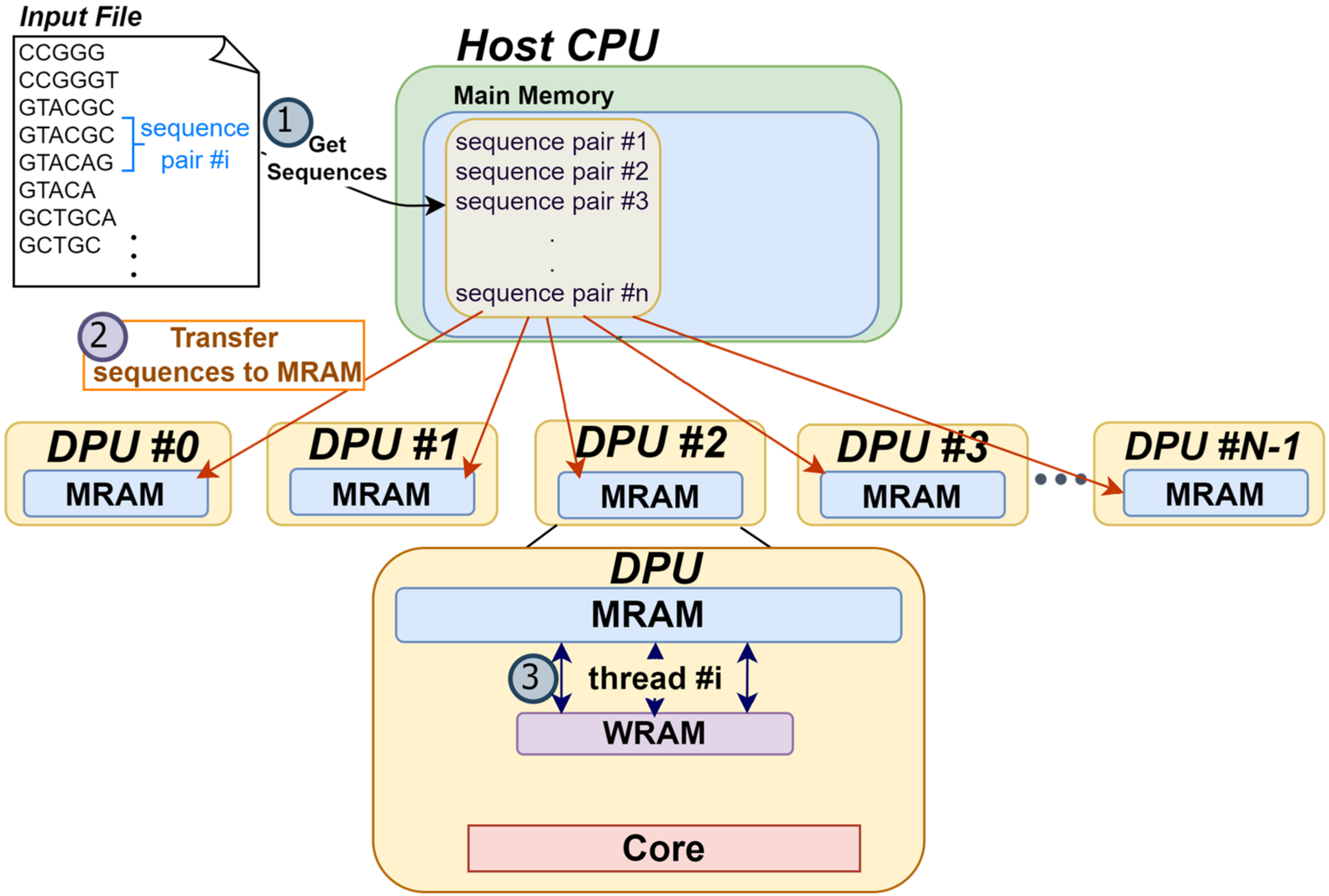
AIM: Alignment-in-Memory



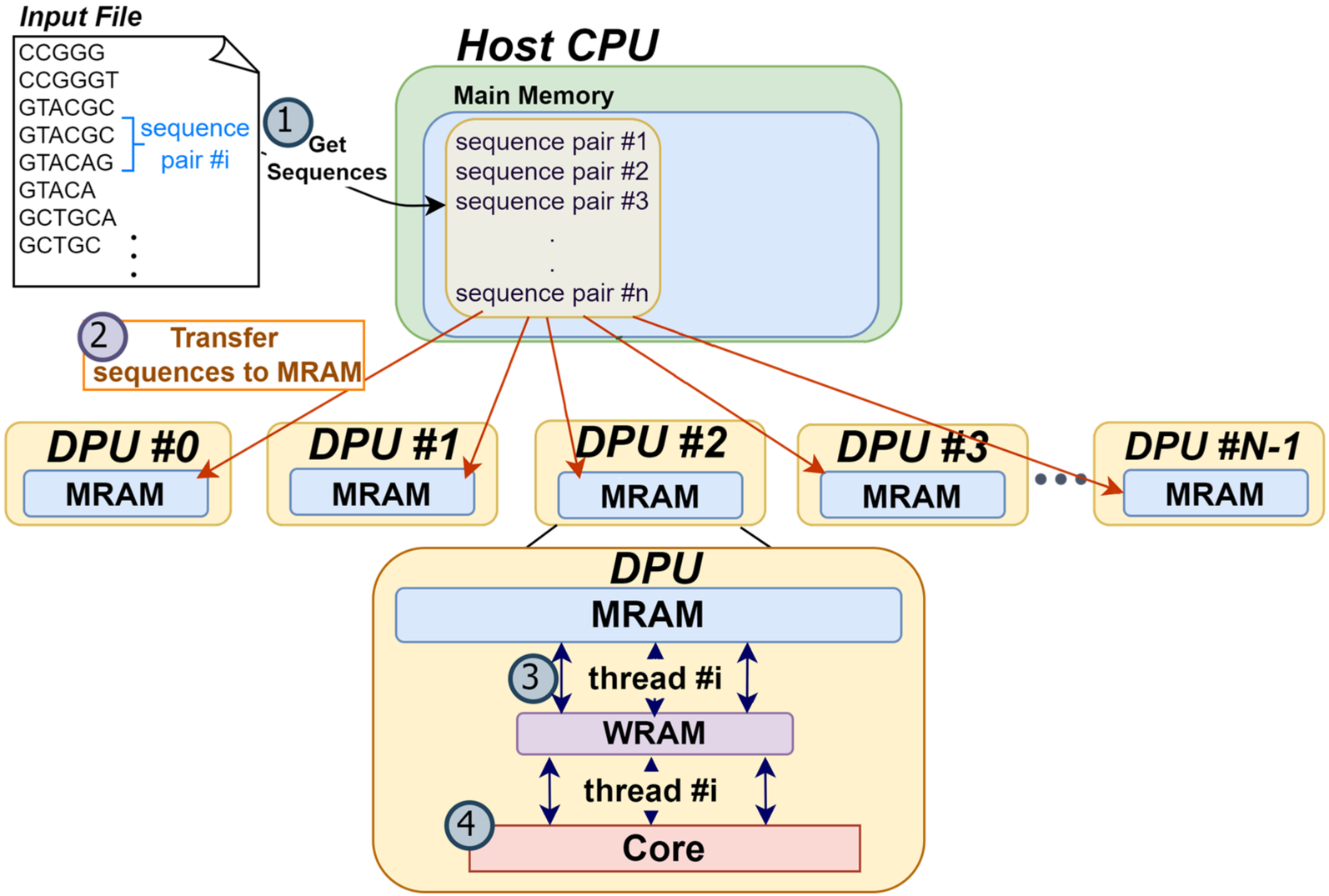
AIM: Alignment-in-Memory



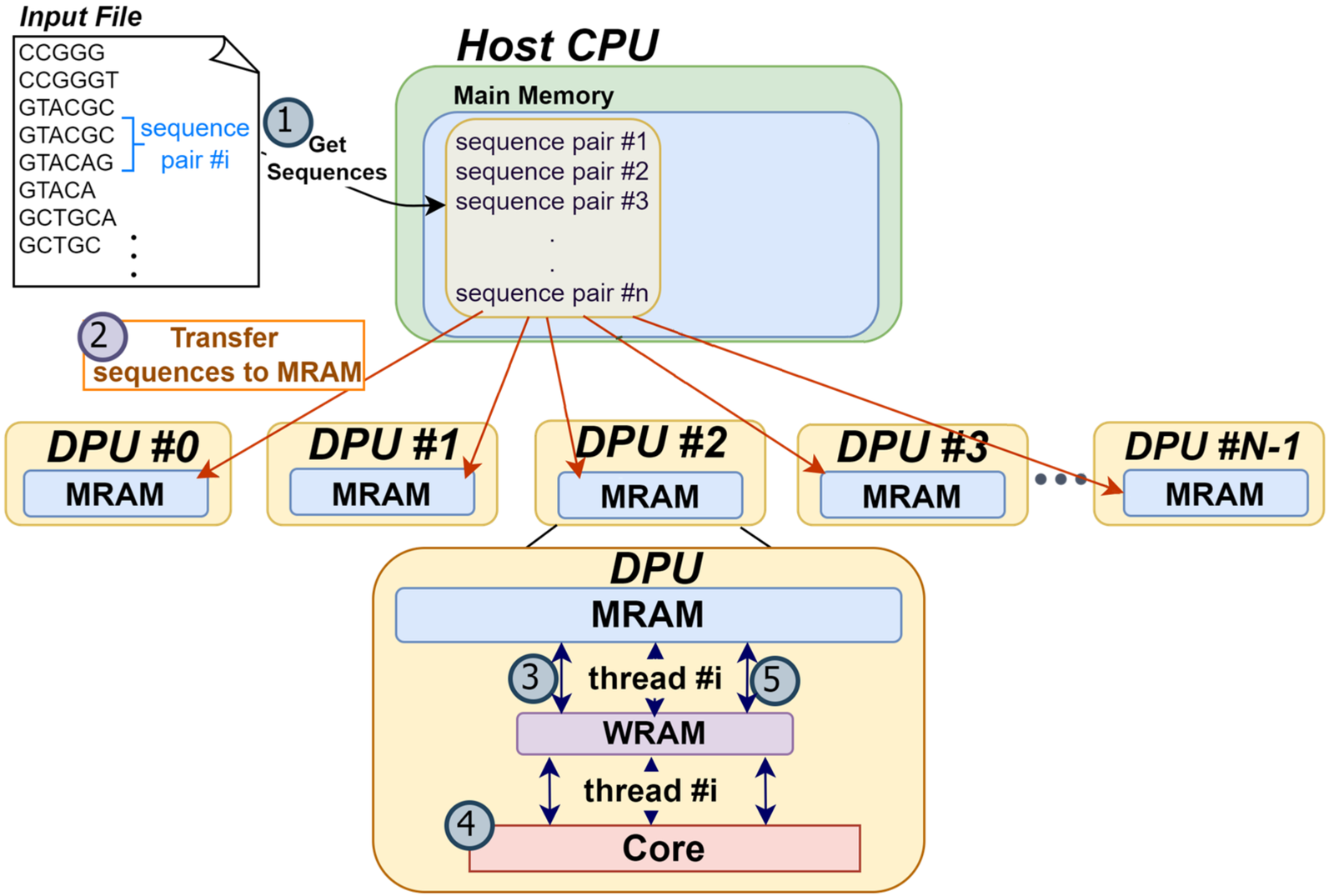
AIM: Alignment-in-Memory



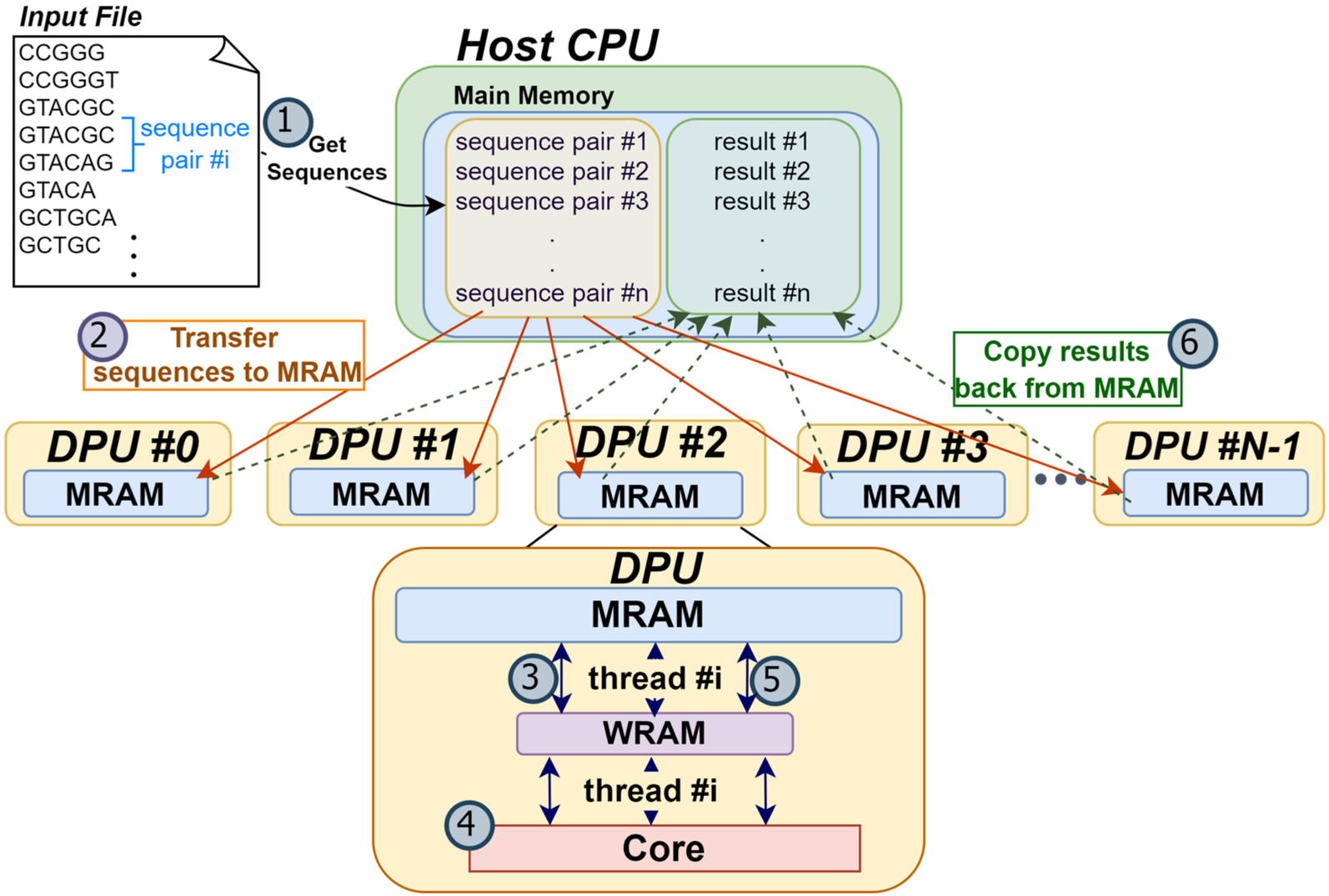
AIM: Alignment-in-Memory



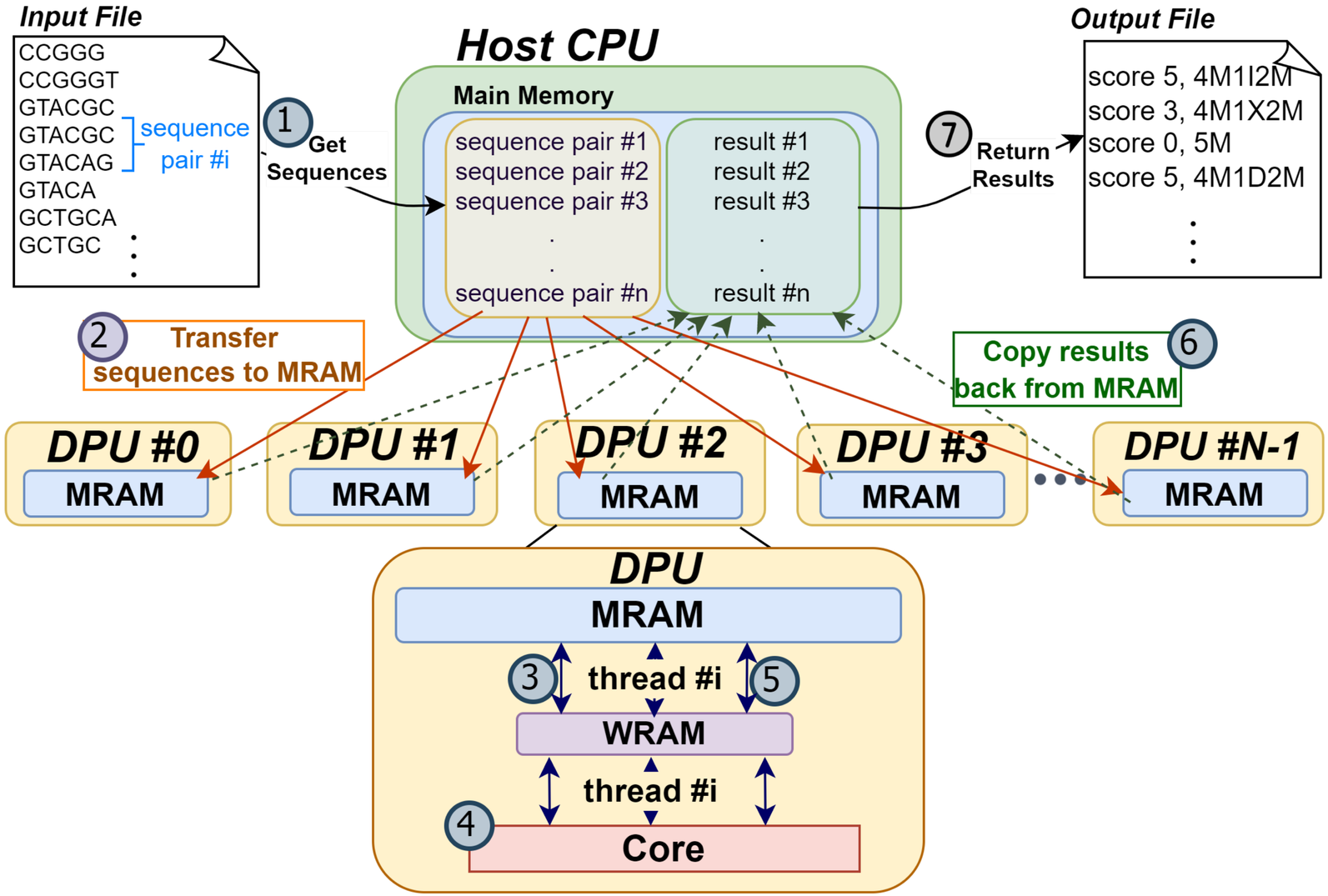
AIM: Alignment-in-Memory



AIM: Alignment-in-Memory



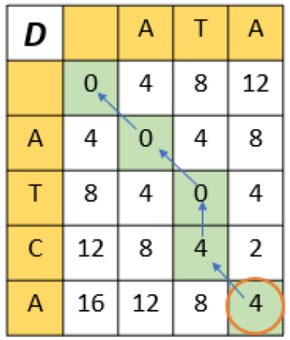
AIM: Alignment-in-Memory



AMERICAN UNIVERSITY OF BEIRUT
FACULTY OF ARTS & SCIENCES

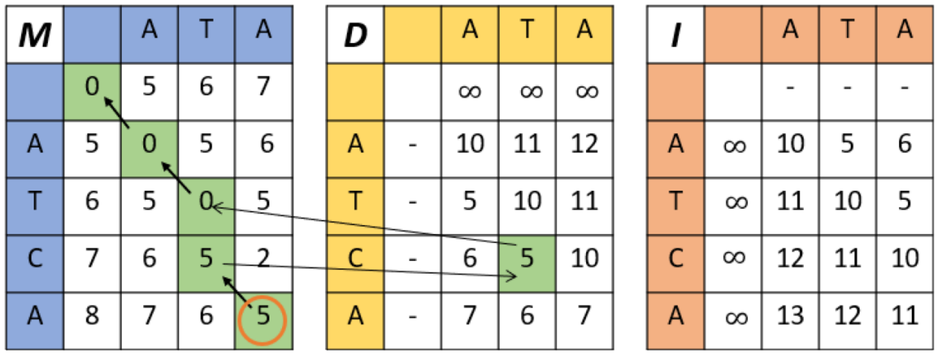
ETH zürich

Supported Algorithms



D		A	T	A
	0	4	8	12
A	4	0	4	8
T	8	4	0	4
C	12	8	4	2
A	16	12	8	4

Needleman-Wunsch (NW)

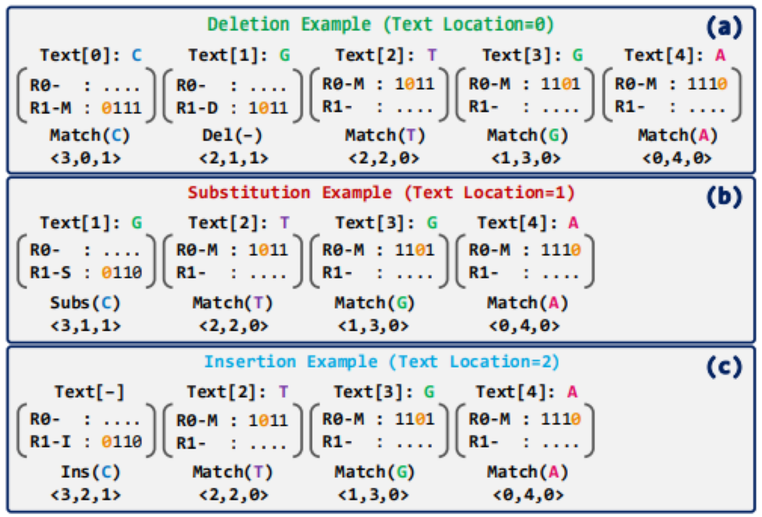


M		A	T	A
	0	5	6	7
A	5	0	5	6
T	6	5	0	5
C	7	6	5	2
A	8	7	6	5

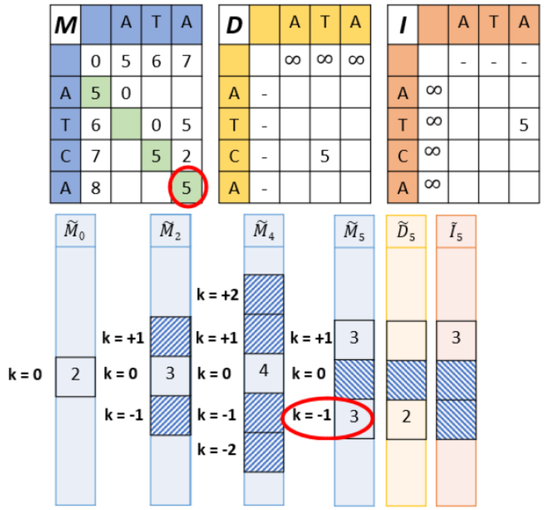
D		A	T	A
		∞	∞	∞
A	-	10	11	12
T	-	5	10	11
C	-	6	5	10
A	-	7	6	7

I		A	T	A
		-	-	-
A	∞	10	5	6
T	∞	11	10	5
C	∞	12	11	10
A	∞	13	12	11

Smith-Waterman-Gotoh (SWG)



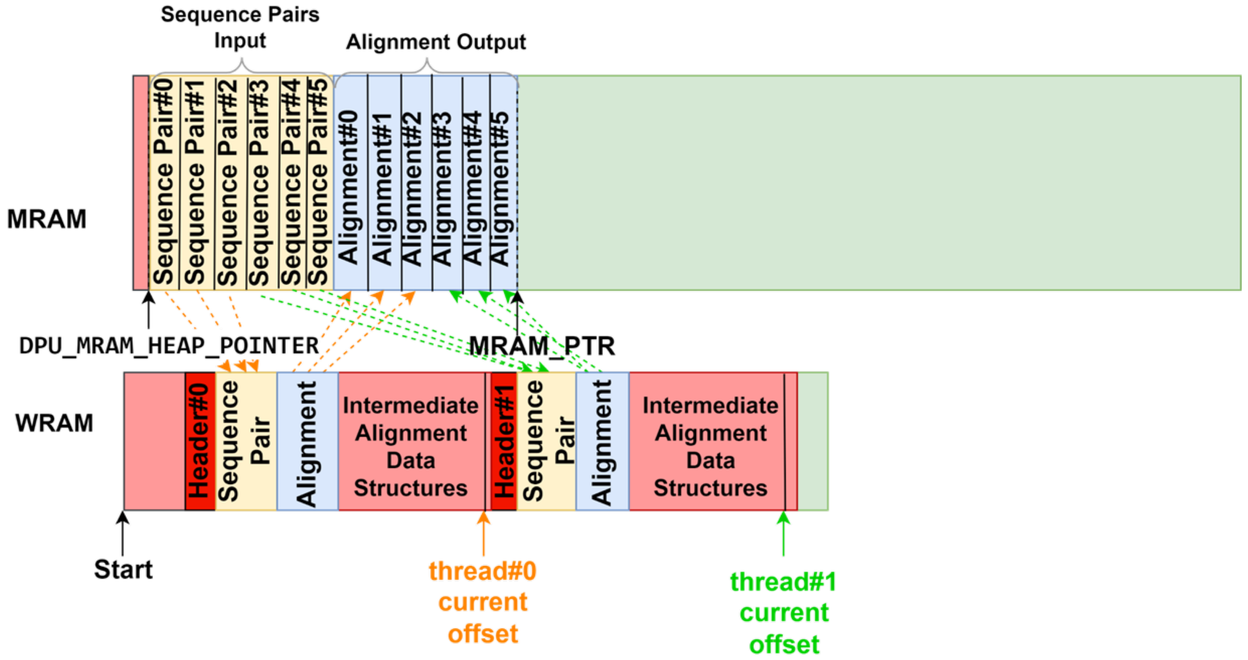
GenASM



Wavefront Algorithm (WFA)

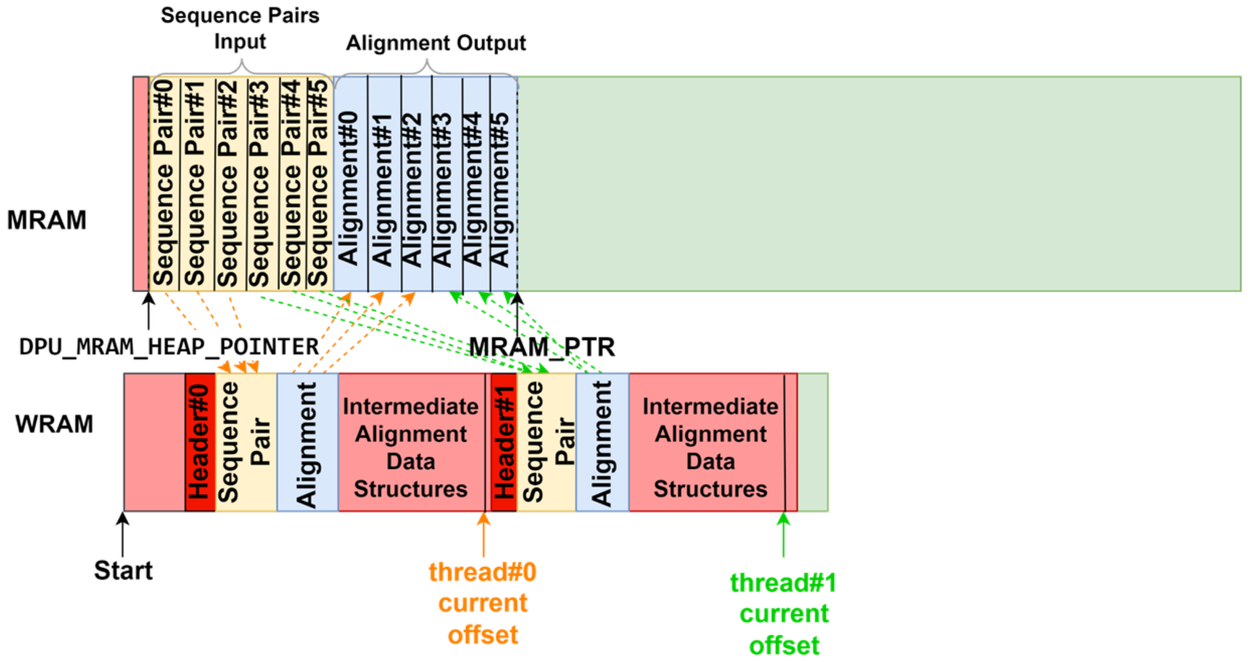
Cali, Damla Senol, et al. "GenASM: A high-performance, low-power approximate string matching acceleration framework for genome sequence analysis." *MICRO*, 2020.
Marco-Sola, Santiago, et al. "Fast gap-affine pairwise alignment using the wavefront algorithm." *Bioinformatics* 37.4 (2021): 456-463.

Managing the UPMEM Memory Hierarchy

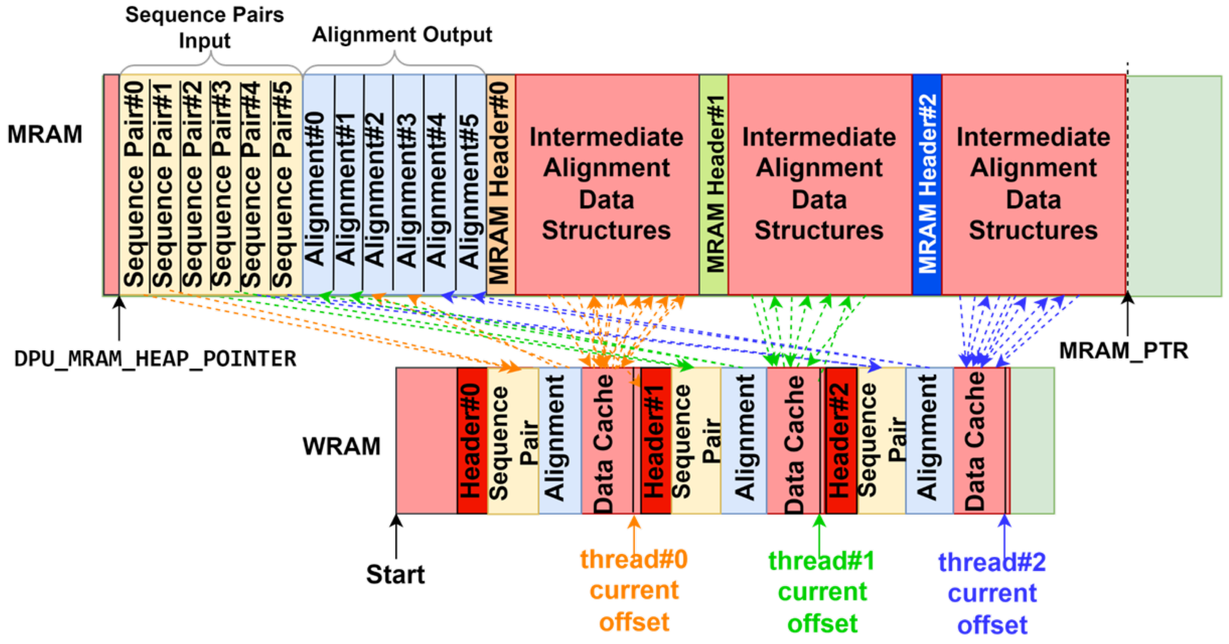


Using WRAM only for intermediate data structures

Managing the UPMEM Memory Hierarchy



Using WRAM only for intermediate data structures



Using WRAM and MRAM for intermediate data structures

Outline

- Background on sequence alignment
- Processing-in-memory
- Our framework
- **Evaluation**

Methodology

- Real and synthetic datasets

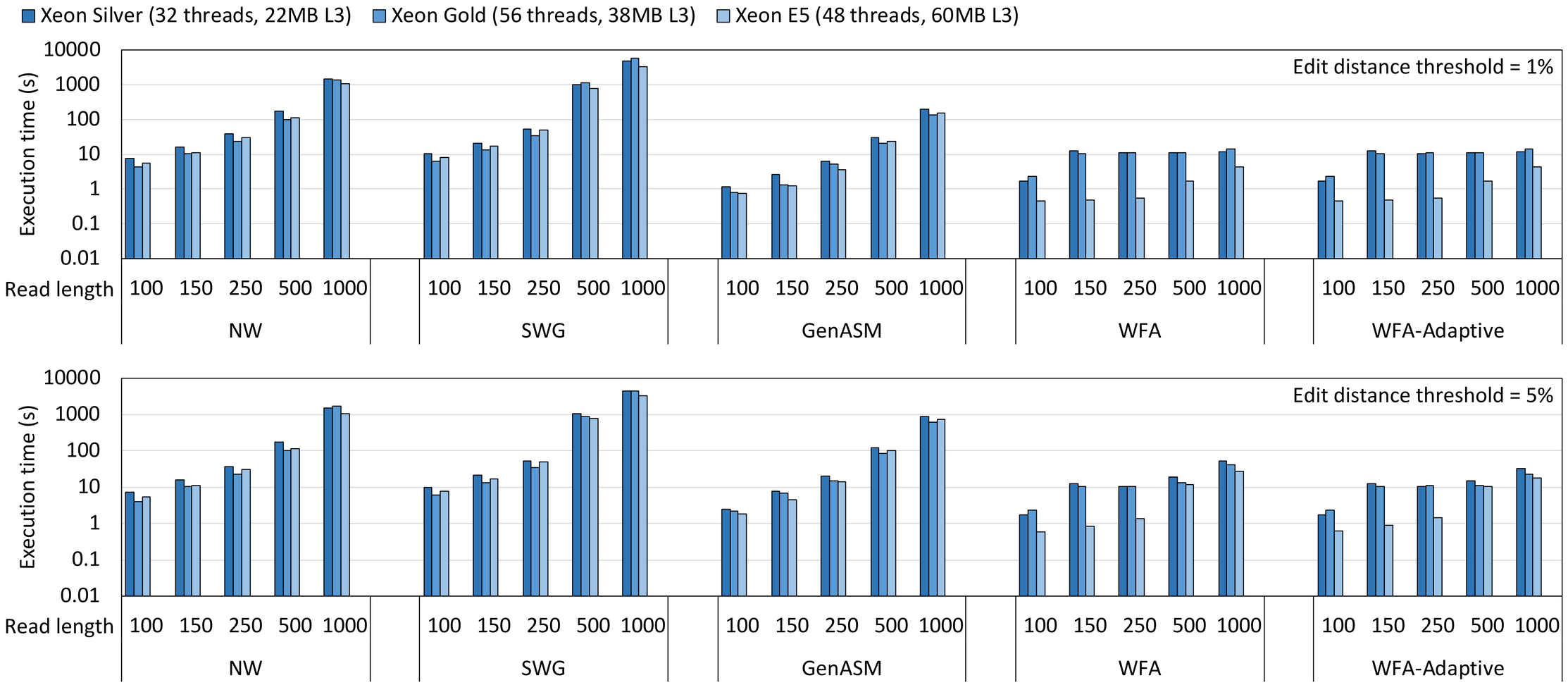
Read Lengths	Edit Distances	Description
100	0-5%	Real, Accession# ERR240727 *
150	0-5%	Real, Accession# SRR826460 *
250	0-5%	Real, Accession# SRR826471 *
500, 1000, 5000, 10000	0-5%	Synthetic ^

*https://www.ebi.ac.uk/ena/browser/view
^https://github.com/smarco/WFA2-lib

- Evaluated systems
  - UPMEM PIM system with 2,560 PIM cores @ 425 MHz and 160 GB of DRAM
  - Three CPU systems

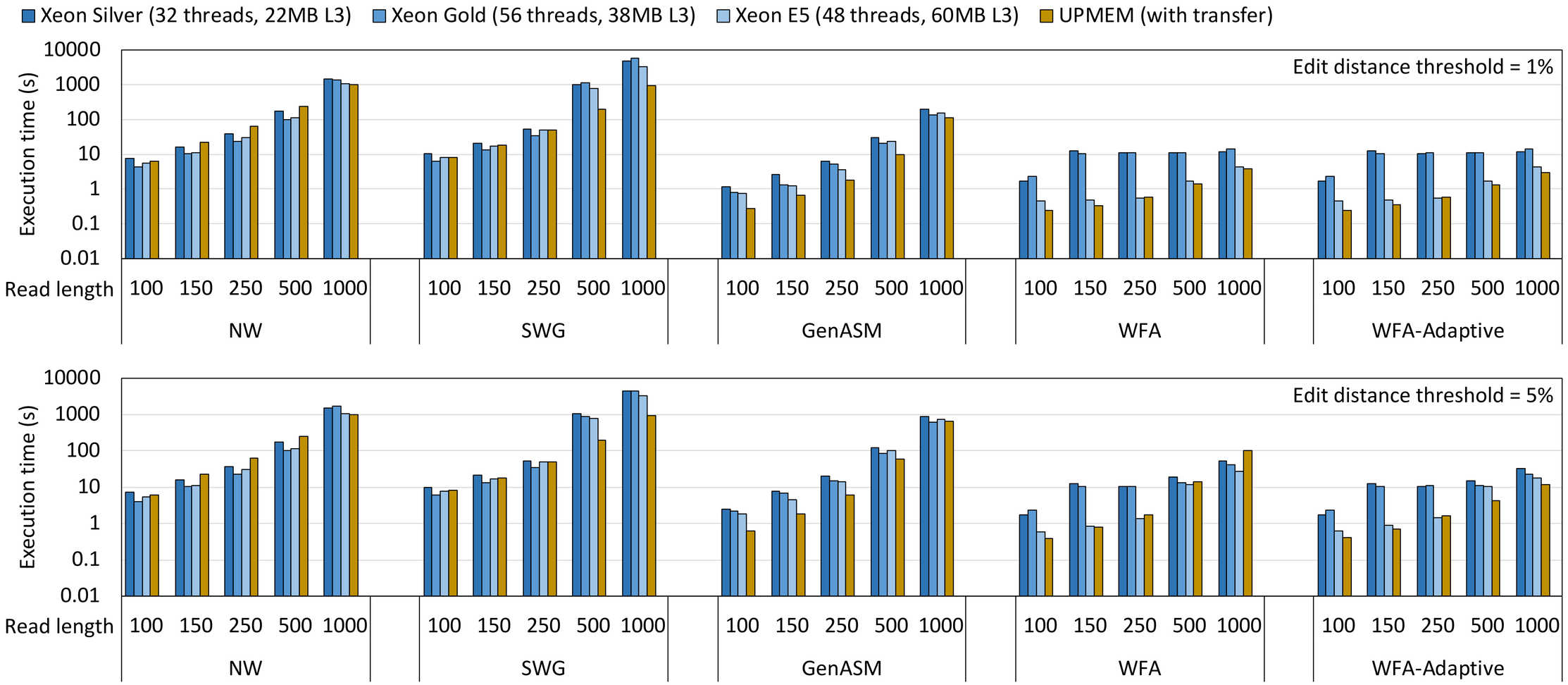
System	1	2	3
CPU	Intel Xeon Silver 4215	Intel Xeon Gold 5120	Intel Xeon E5-2697 v2
Process node	14 nm	14 nm	22 nm
Sockets	2	2	2
Cores	16	28	24
Threads	32	56	48
Frequency	2.50 GHz	2.20 GHz	2.70 GHz
L3 cache	22 MB	38 MB	60 MB
Memory	256 GB	64 GB	32 GB
CPU TDP	170 W	210 W	260 W

PIM vs CPU



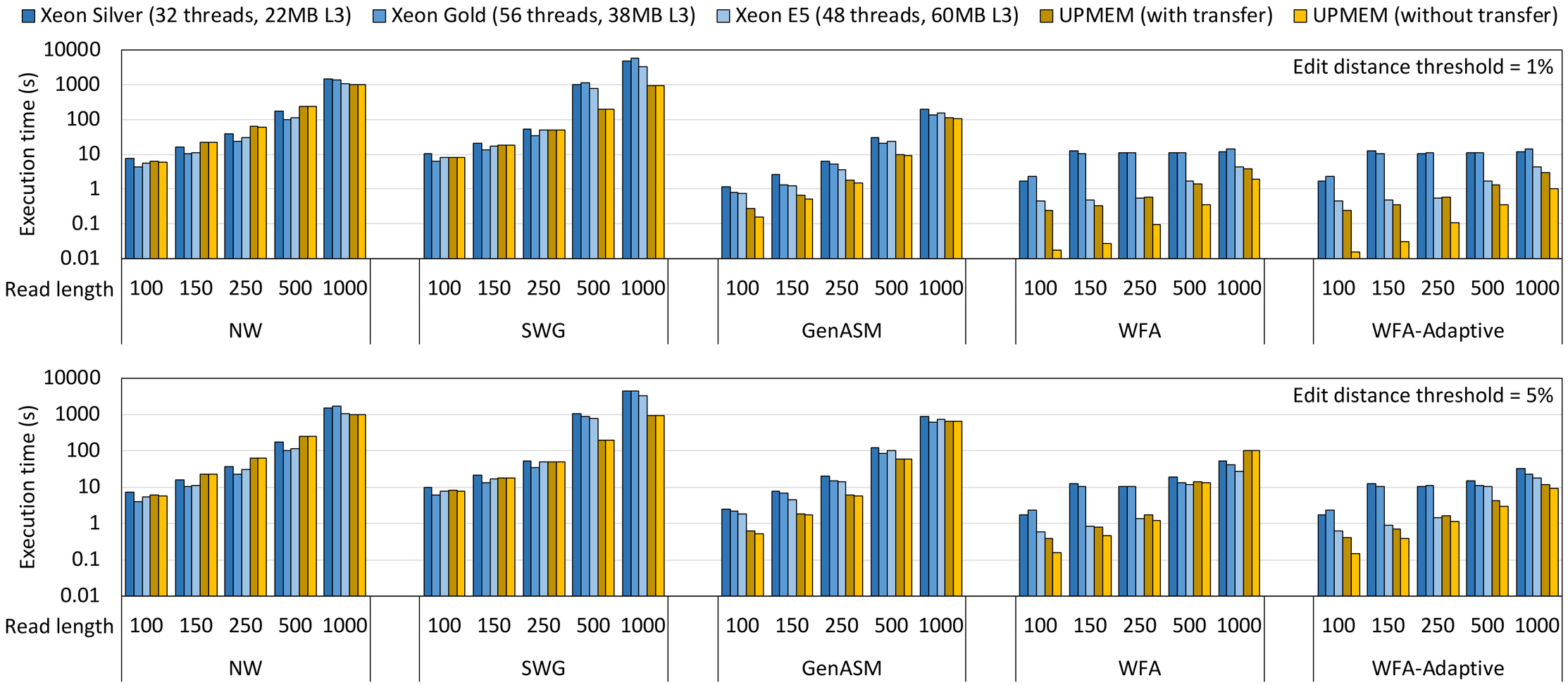
Observation #1: The best performing CPU system is the one with the largest L3 cache (demonstrates memory-boundedness of sequence alignment)

PIM vs CPU

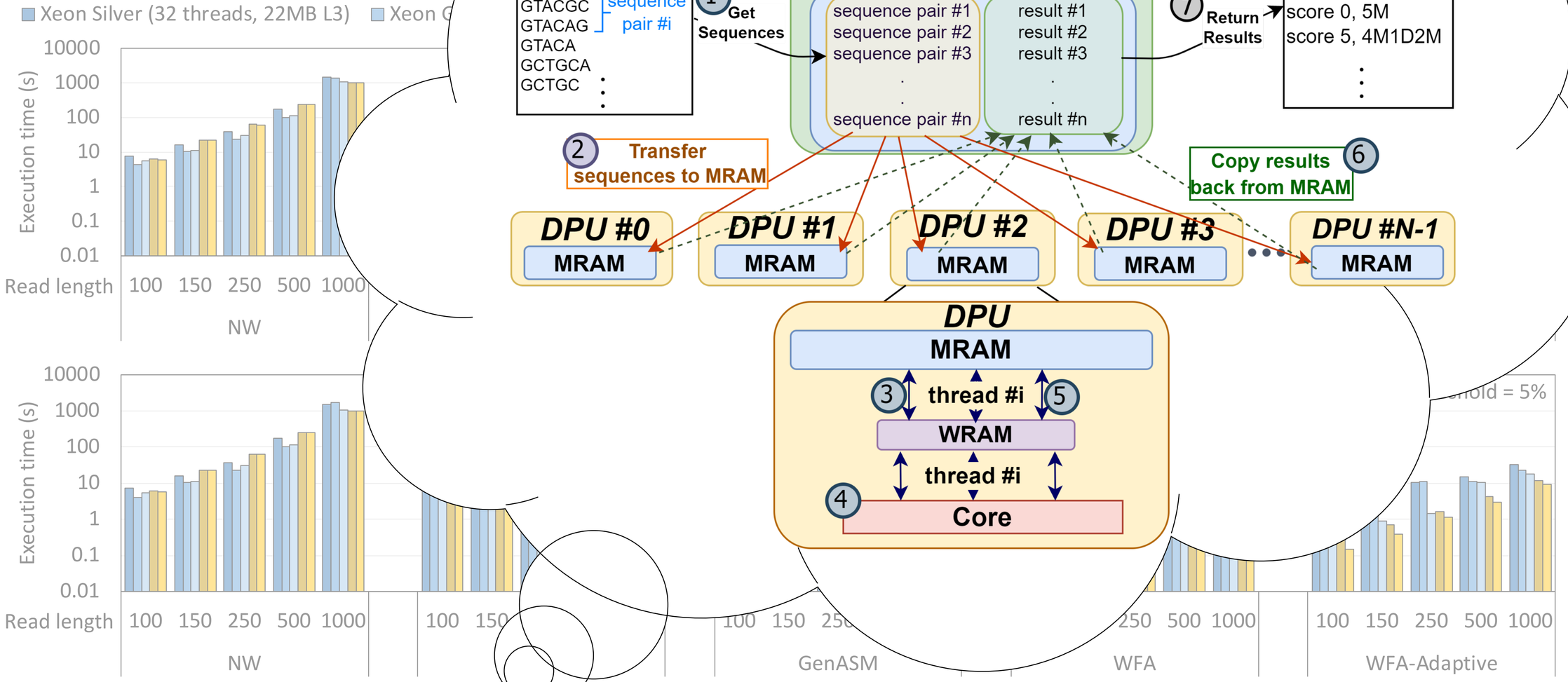


Observation #2: PIM outperforms CPU in the majority of cases
(up to 4.06× for SWG, up to 1.83× for WFA, and up to 2.56× for WFA-adaptive)

PIM vs CPU

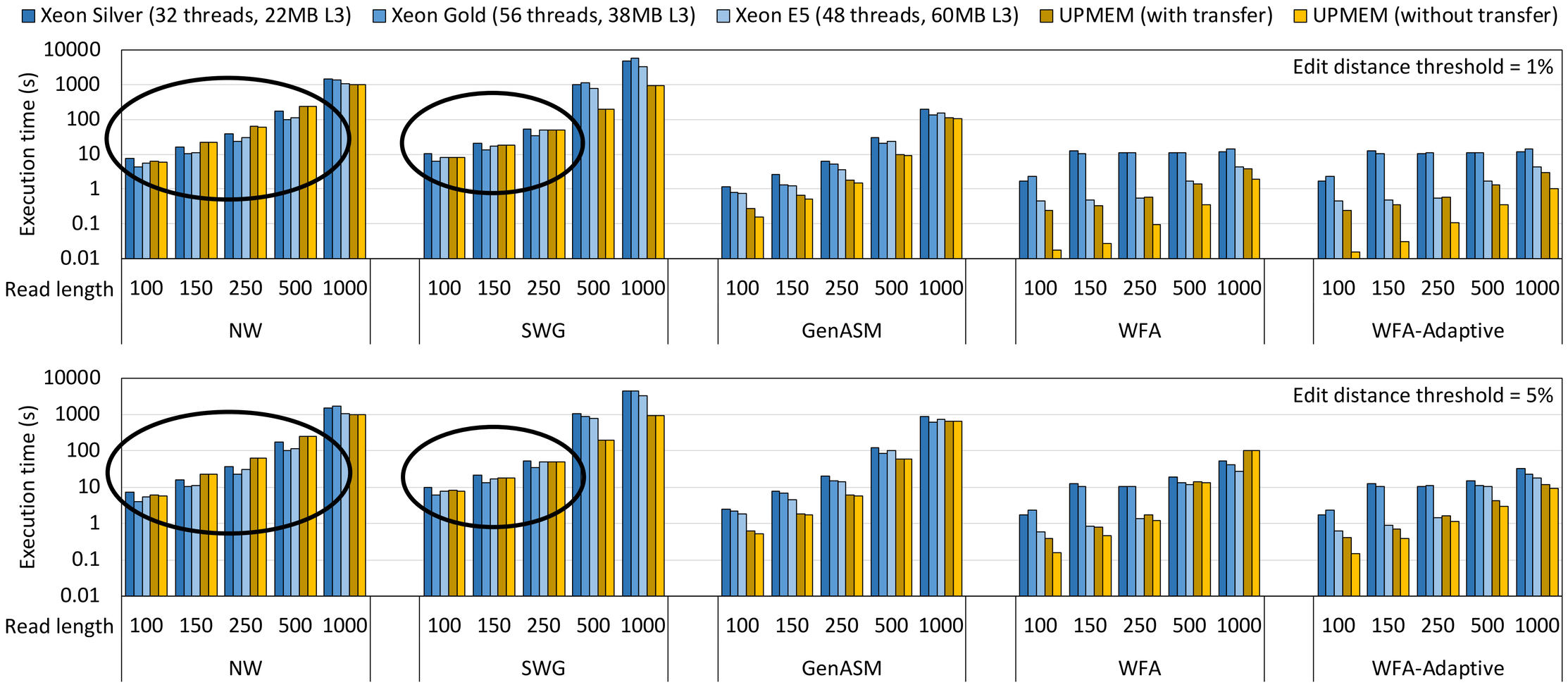


Observation #3: When data transfer time is not included, PIM outperforms CPU even more (up to 25.93× for WFA, up to 28.14× for WFA-adaptive)



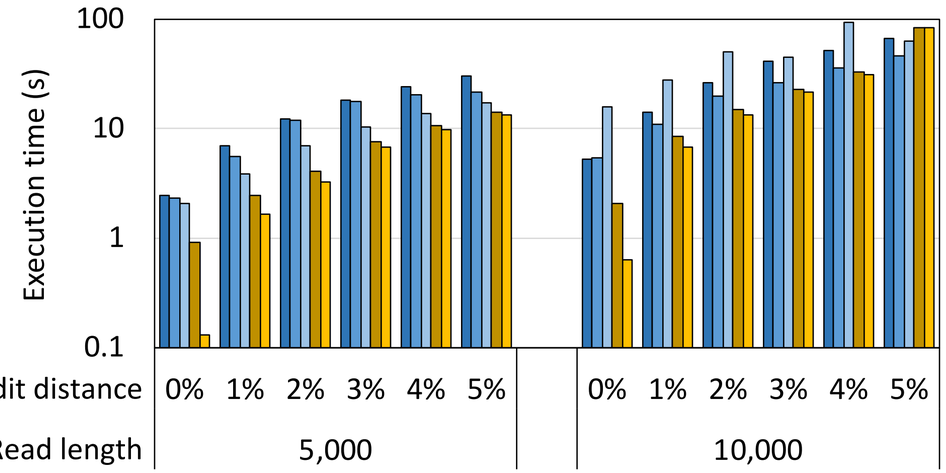
Observation #3: When data transfer time is not included, PIM outperforms CPU even more (up to 25.93× for WFA, up to 28.14× for WFA-adaptive)

PIM vs CPU



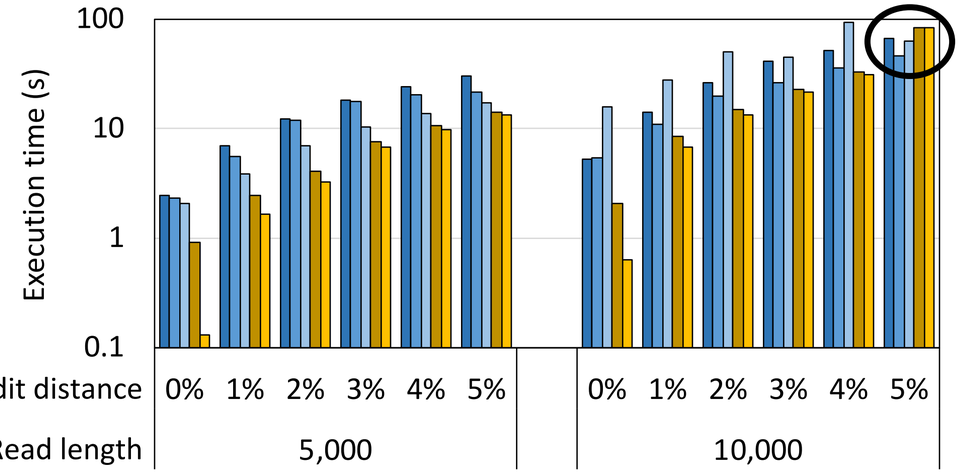
Observation #4: PIM does not outperform CPU for algorithms with regular access patterns at small read lengths

PIM vs. CPU for WFA-adaptive with Large Read Lengths



Observation #1: PIM continues to outperform CPU for very large read lengths

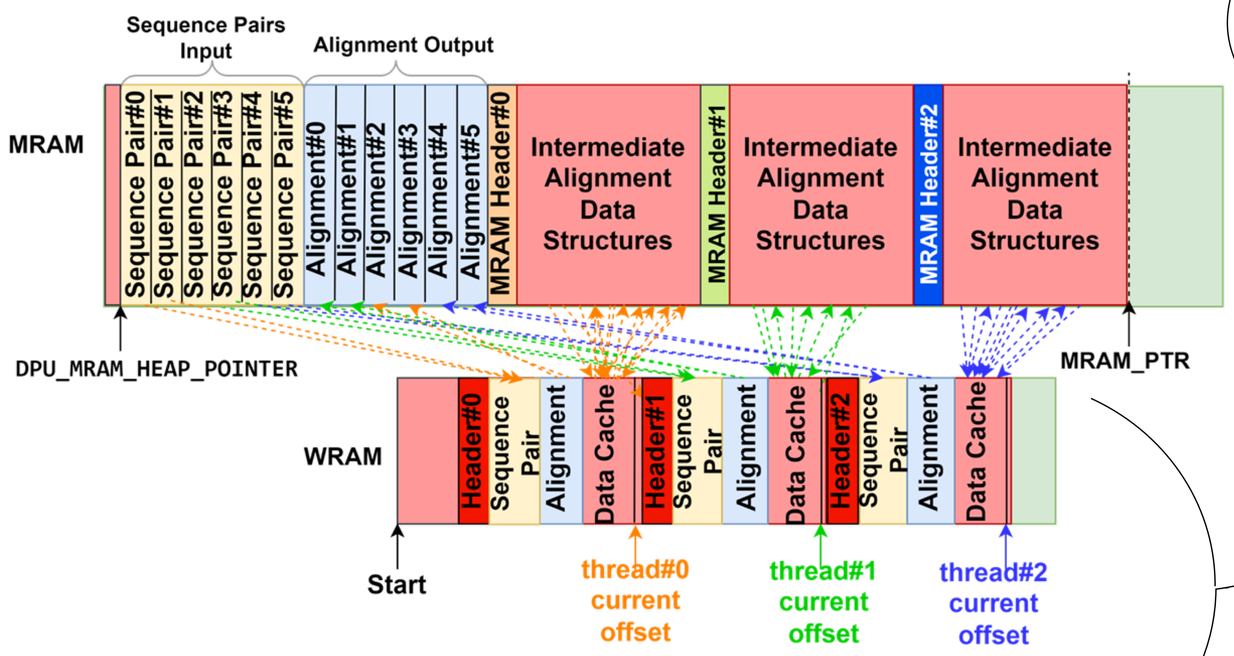
PIM vs. CPU for WFA-adaptive with Large Read Lengths

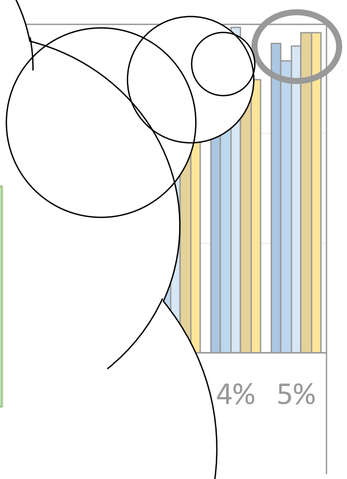


Observation #1: PIM continues to outperform CPU for very large read lengths

Observation #2: Scalability currently limited by WRAM capacity

PIM vs. CPU for WFA-adaptive with Large Read Lengths





ery large read lengths

ted by WRAM capacity

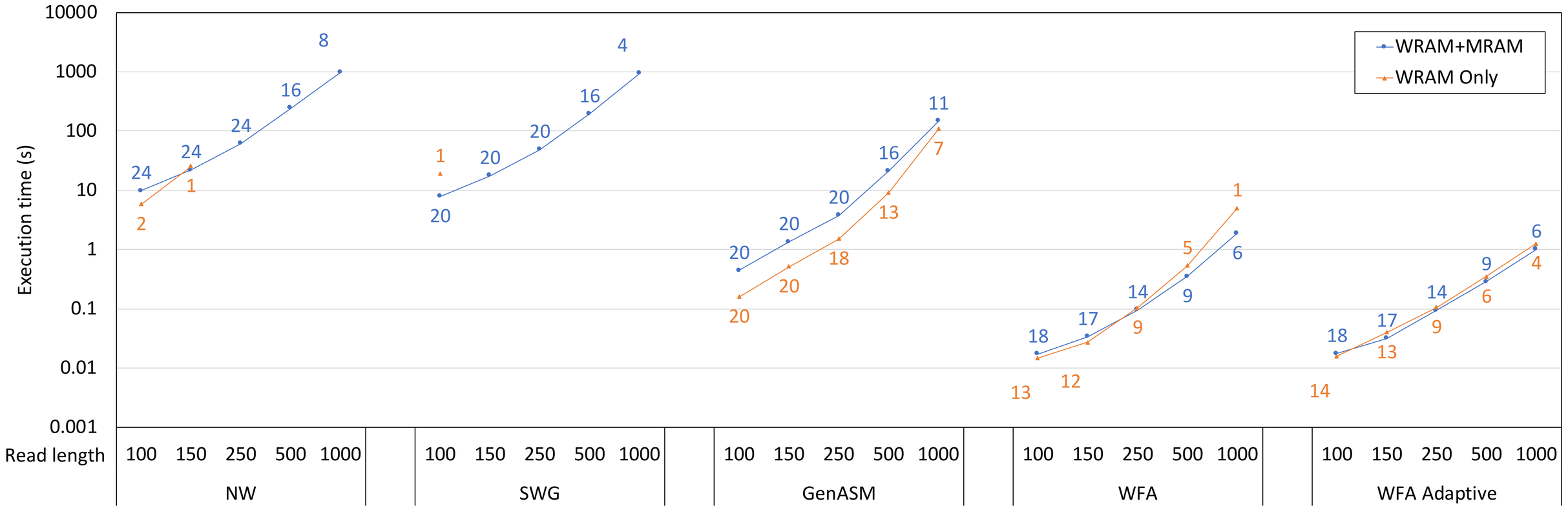
PIM vs. GPU

Sequence length	Edit distance	Throughput (alignments per second)		Throughput improvement
		WFA-GPU	UPMEM (with transfer)	
150	2%	9.09M	12.97M	1.42×
	5%	5.56M	7.03M	1.27×
1,000	2%	1.43M	1.10M	0.77×
	5%	370K	434K	1.17×
10,000	2%	25.0K	66.9K	2.68×
	5%	5.56K	11.81K	2.12×

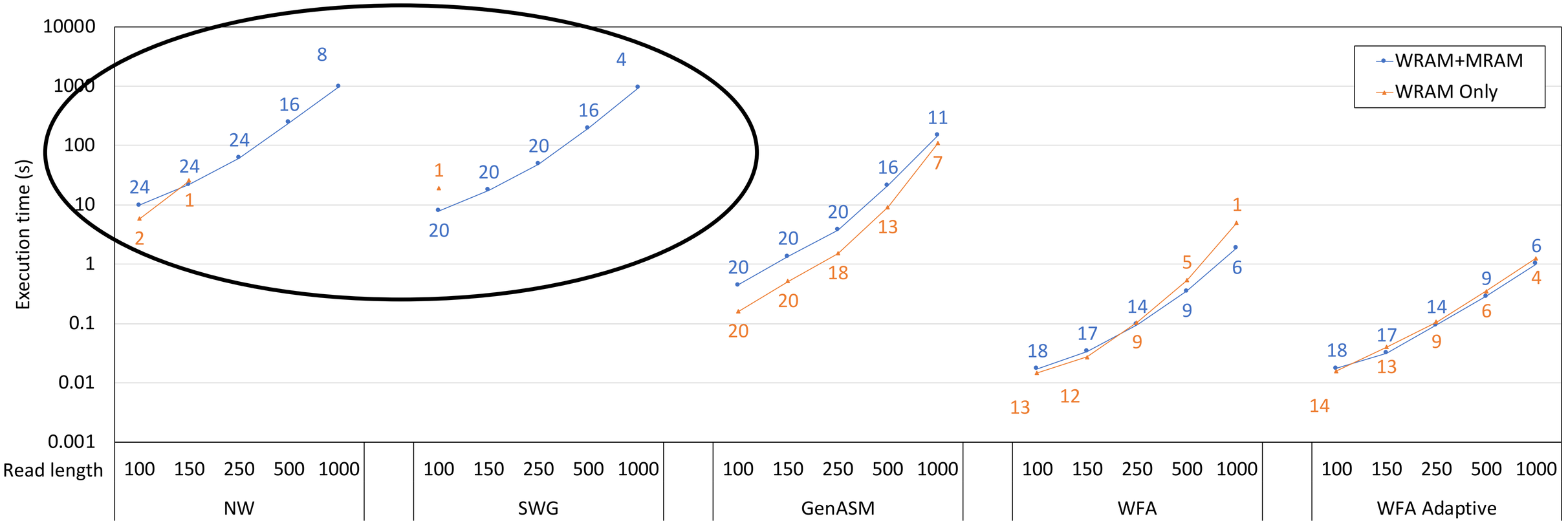
Observation: PIM outperforms GPU in the majority of cases

Aguado-Puig, Quim, et al. "WFA-GPU: Gap-affine pairwise alignment using GPUs." bioRxiv (2022)

Using WRAM only vs. WRAM and MRAM for Intermediate Data Structures

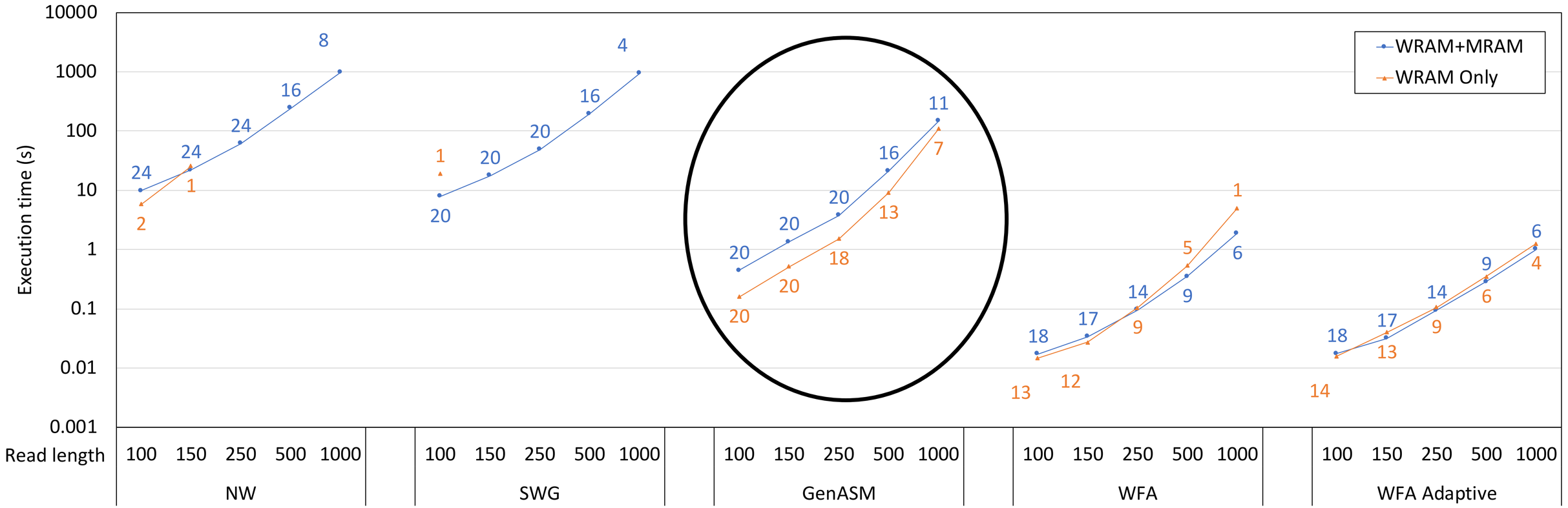


Using WRAM only vs. WRAM and MRAM for Intermediate Data Structures



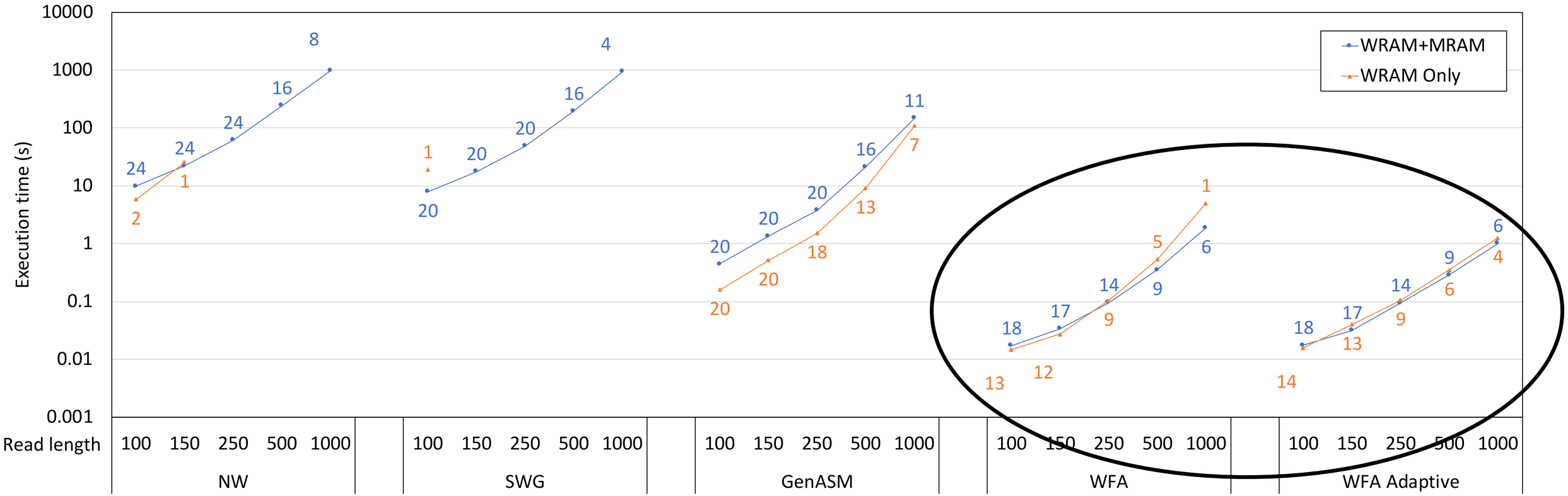
Observation #1: For algorithms that use large data structures (NW and SWG), WRAM only does not scale

Using WRAM only vs. WRAM and MRAM for Intermediate Data Structures



Observation #2: For algorithms that use small data structures (GenASM), WRAM only is better

Using WRAM only vs. WRAM and MRAM for Intermediate Data Structures



Observation #3: For algorithms that use medium-sized data structures (WFA, WFA-adaptive), WRAM only is better for short reads while WRAM+MRAM is better for long reads

Summary

- Sequence alignment is **memory-bound** on traditional processor-centric systems
- **Processing-in-memory (PIM)** overcomes the memory bandwidth bottleneck by placing cores near the memory
- We present **Alignment-in-Memory (AIM)**, a framework for sequence alignment on real PIM systems
  - Supports multiple alignment algorithms: NW, SWG, GenASM, and WFA
  - Implemented on UPMEM, the first real PIM system
- Results show **substantial speedups over CPUs and GPUs**
- AIM is available at: https://github.com/safaad/aim