Accelerating the Phylogenetic Likelihood Function using Versal Adaptive SoCs

Geert Roks, Mario Ruiz Noguera (AMD), Nikolaos Alachiotis (UT)

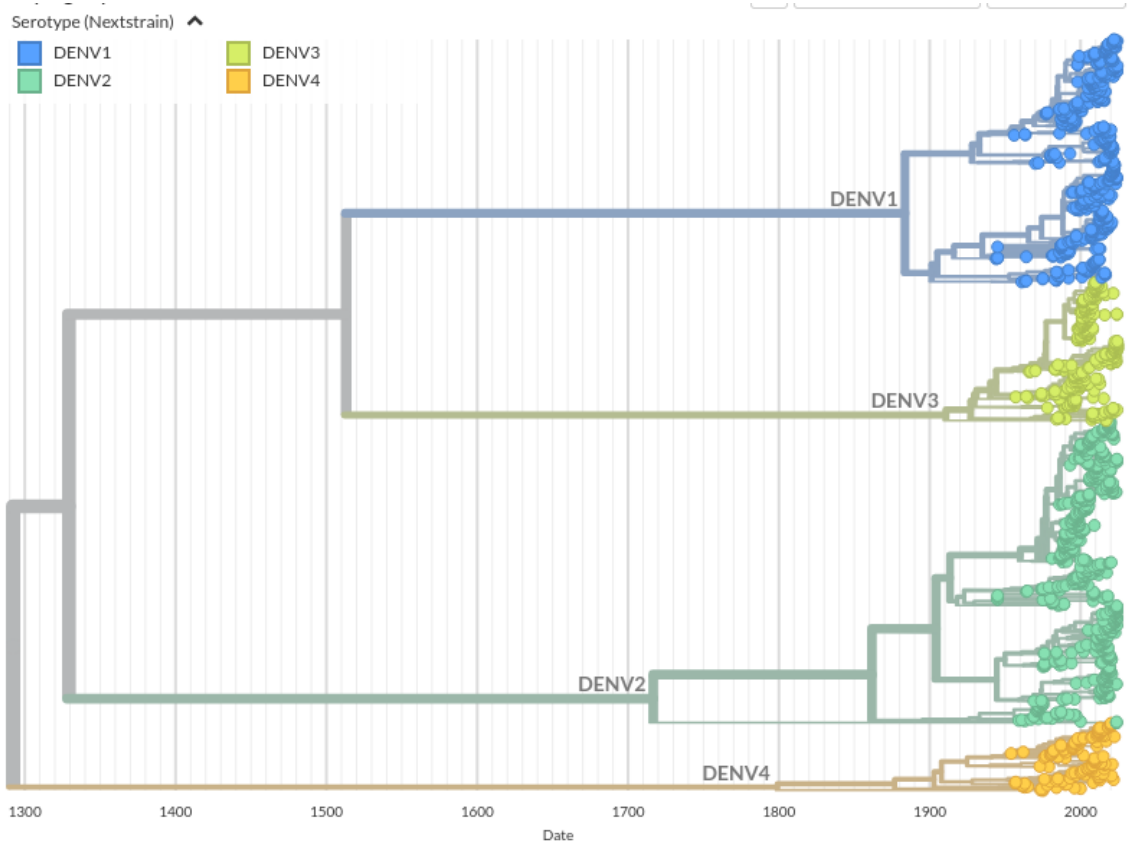
University of Twente

Contents

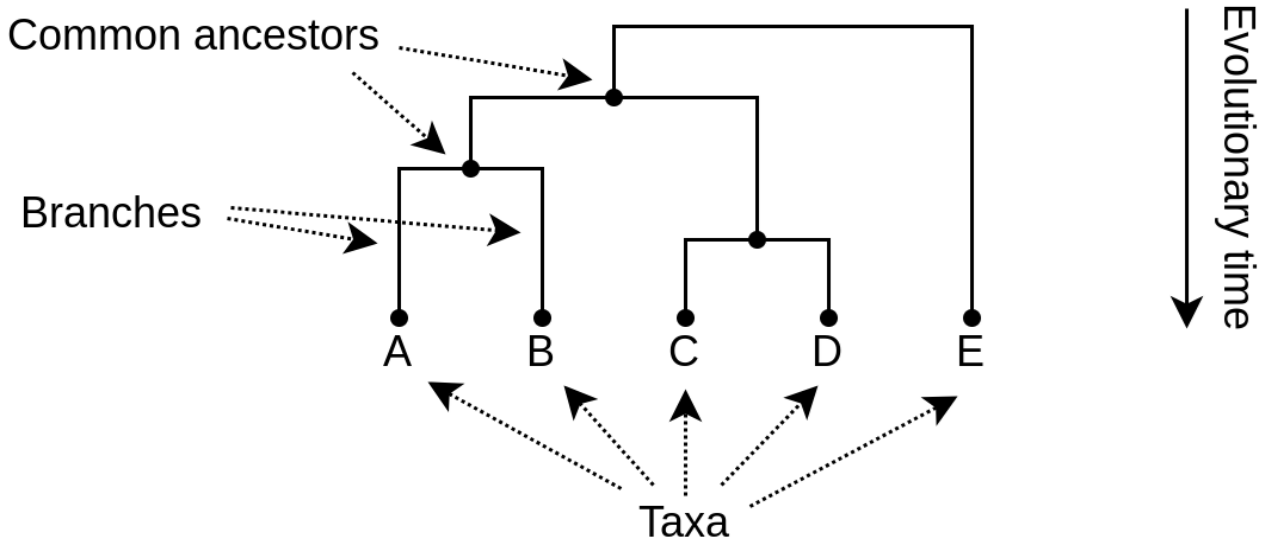
Introduction

Phylogenetics

- Study of evolutionary history
- Relationship between organism
- Epidemiology



Phylogenetic tree



Phylogenetic Analysis

Sequence1

Sequencing

Sequence 2

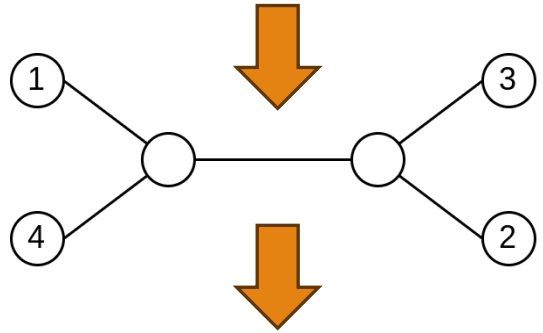
Sequence 3

Sequence 4

Raw data

Alignment

Sequence 1	C	A	T	G	C	G	A	G	T	A	–	G	T	A	G
Sequence 2	C	A	T	G	–	–	–	G	T	A	–	G	T	A	G
Sequence 3	C	C	T	G	–	G	A	G	T	A	C	G	T	A	G
Sequence 4	C	A	T	G	–	–	A	G	–	–	C	G	T	A	G



Scoring Function

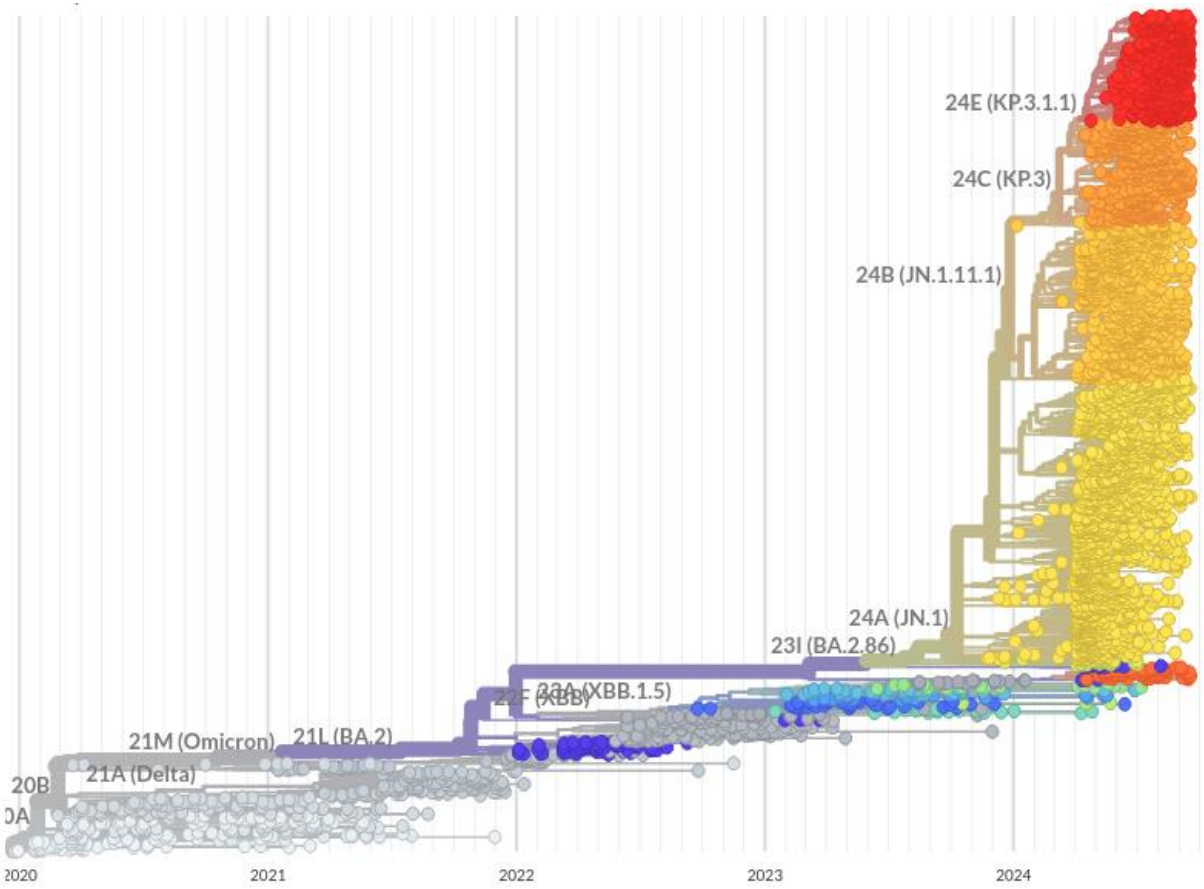
Phylogenetic Likelihood Function (PLF)

- Maximum Likelihood
- Bayesian Inference

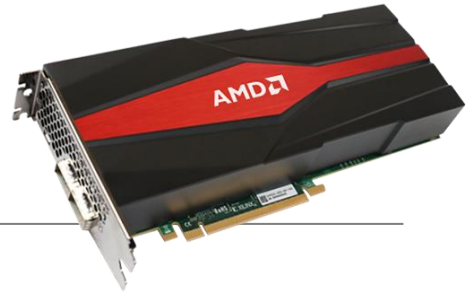
Optimization algorithm

Phylogenetic Likelihood Function Acceleration

- RAxML
  - PLF takes up to 95% of analysis time
- Improved sequencing techniques
  - larger phylogenetic trees
  - need for acceleration
- PLF main parts:
  - Matrix multiplication
  - scaling
- Acceleration efforts:
  - CPU
  - GPU
  - FPGA

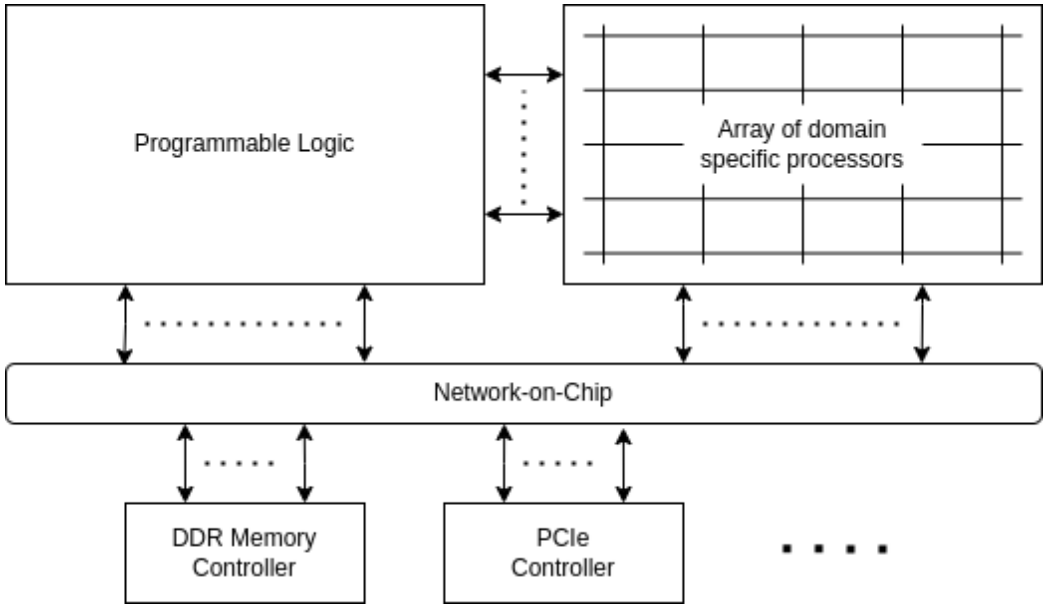


Versal Adaptive SoC



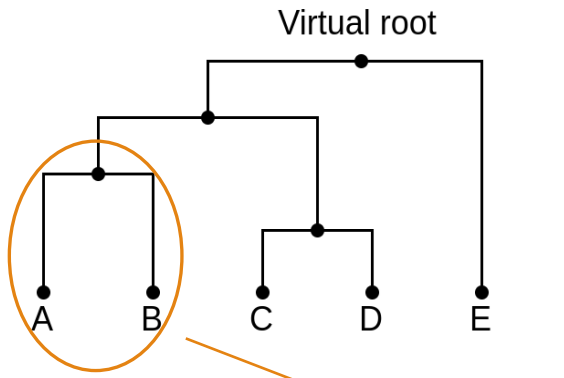
- Tight integration between:
  - Programmable Logic (PL)
  - Array of AI Engines (AIE)

- Idea:
  - AIE array: matrix multiplication
  - PL: scaling and data organization



Background – Phylogenetic Likelihood Function

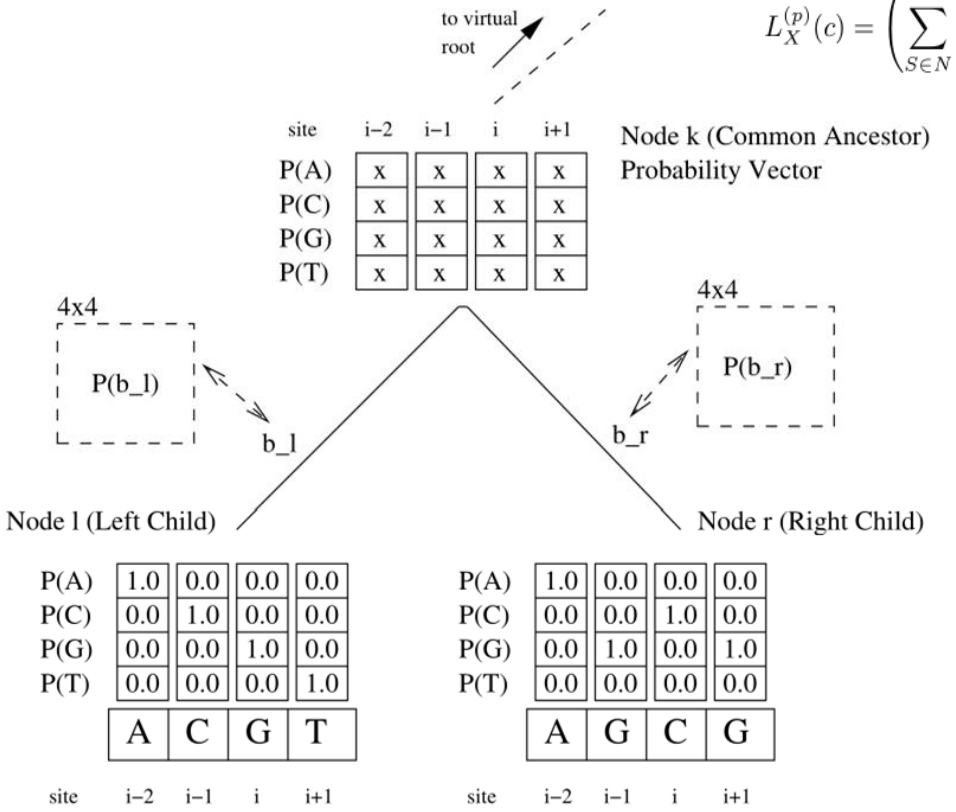
Phylogenetic Likelihood Function



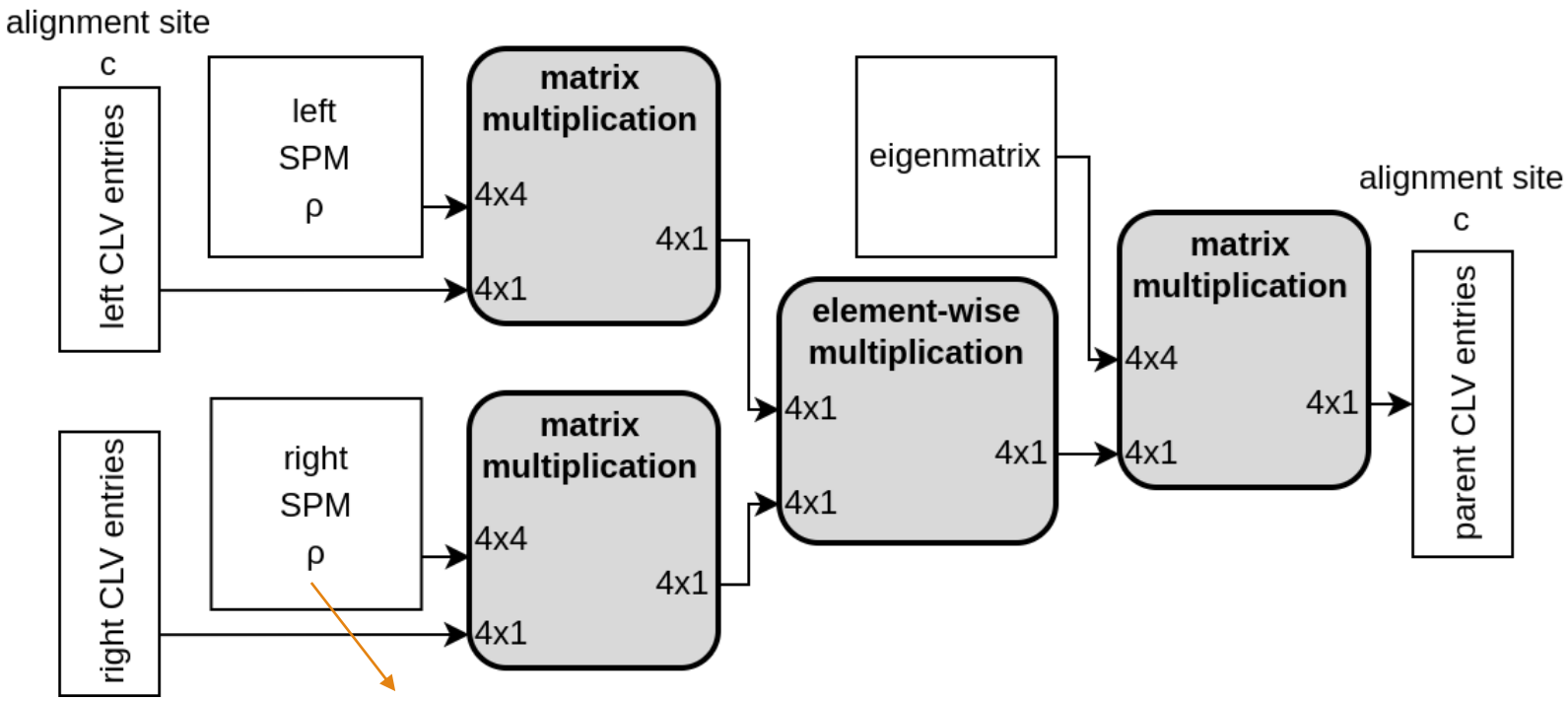
Conditional Likelihood Vector (CLV)

Nucleotide letters

$$L_X^{(p)}(c) = \left(\sum_{S \in N} P_{XS}(t_l, \rho) L_S^{(l)}(c)\right)\left(\sum_{S \in N} P_{XS}(t_r, \rho) L_S^{(r)}(c)\right)$$

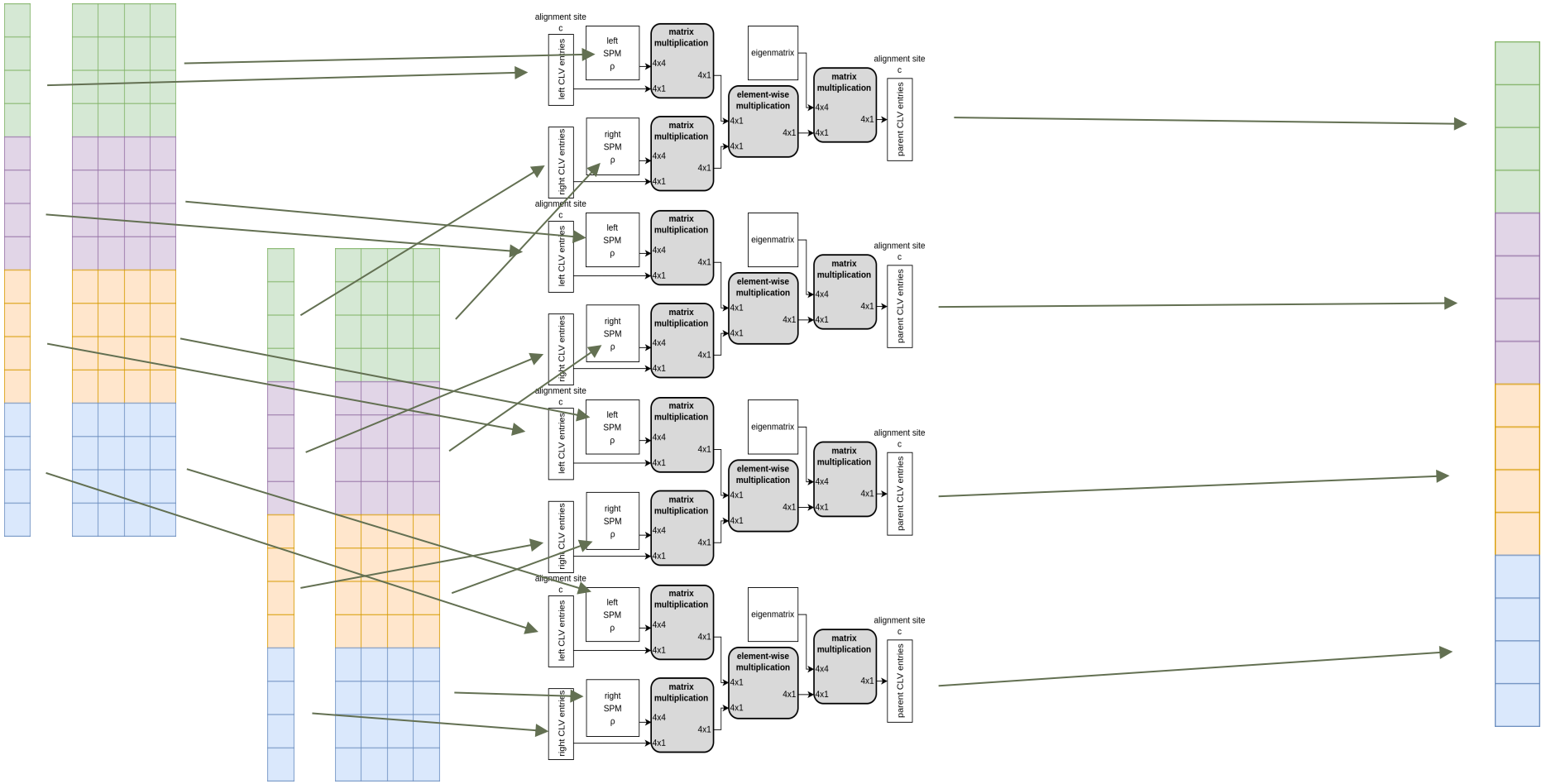


Computational dataflow

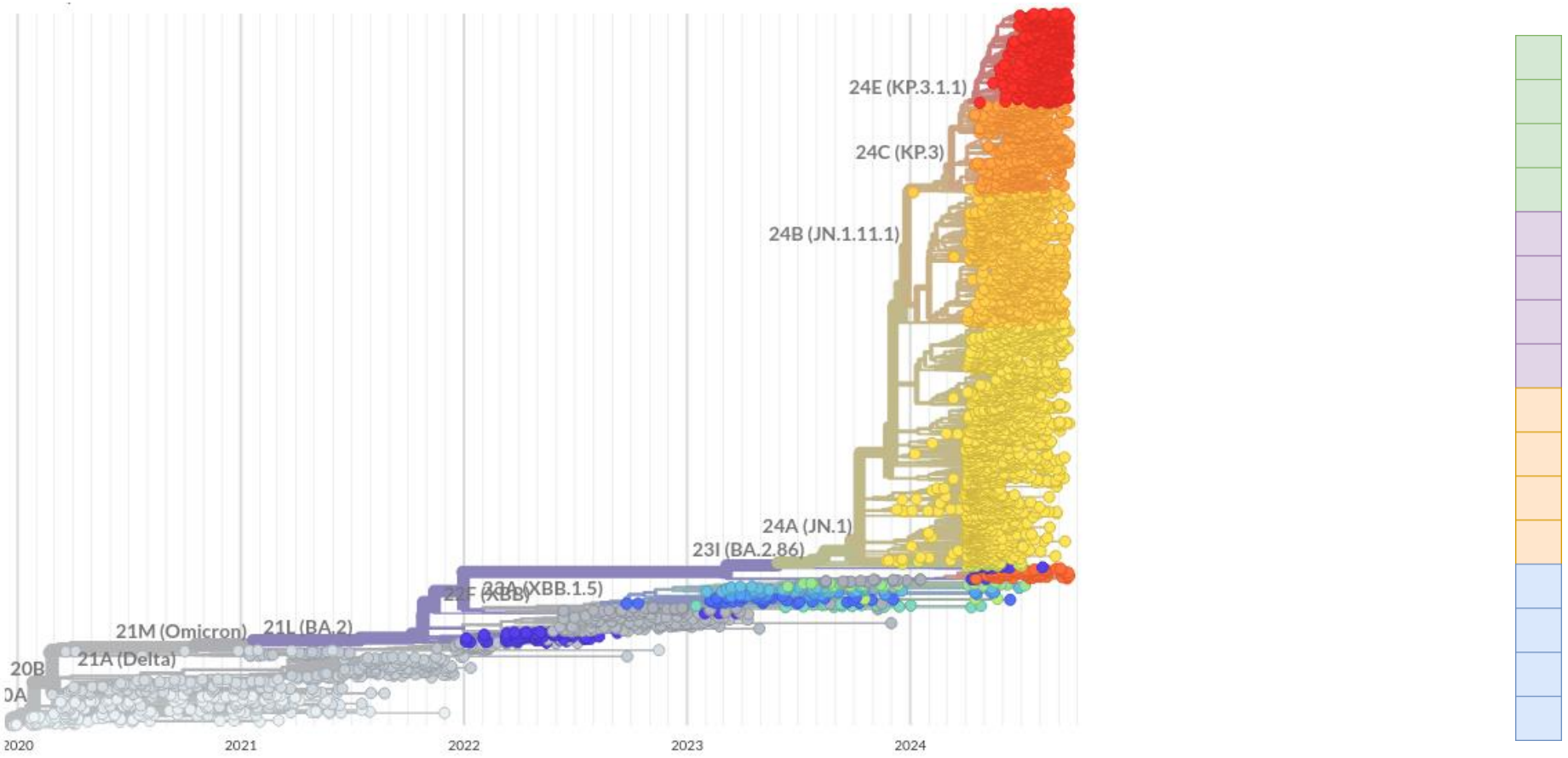


Substitution Probability Matrix / branch matrix

Gamma rates


alignment site
c
left CLV entries
left SPM ρ
right CLV entries
right SPM ρ
matrix multiplication
4x4
4x1
eigenmatrix
element-wise multiplication
parent CLV entries

Scaling


24E (KP.3.1.1)
24C (KP.3)
24B (JN.1.11.1)
24A (JN.1)
23I (BA.2.86)
23A (XBB.1.5)
21M (Omicron)
21L (BA.2)
21A (Delta)
2020
2021
2022
2023
2024

Scaling

If all values in a CLV entry are below threshold ϵ,

then multiply all values in the CLV entry with E

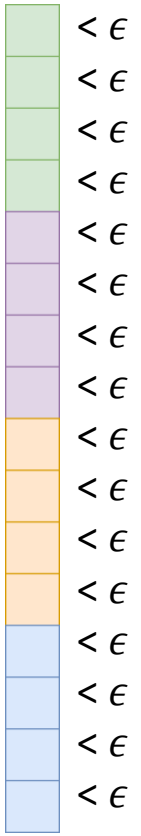
For single precision:

$$\mathrm{E} = 2^{32}$$

$$\epsilon = \frac{1}{\mathrm{E}} = 2^{-32}$$

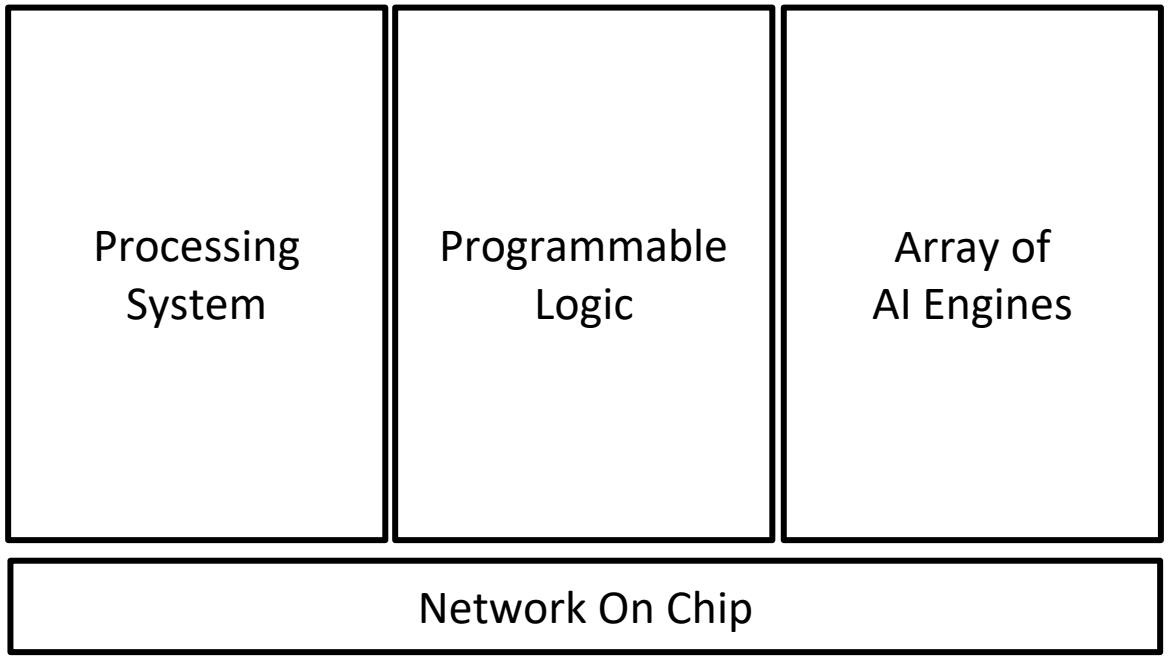
Number of scaling events stored in vector U

U:	6	45	17	105	23	13	...	9



Background – Versal Adaptive SoC

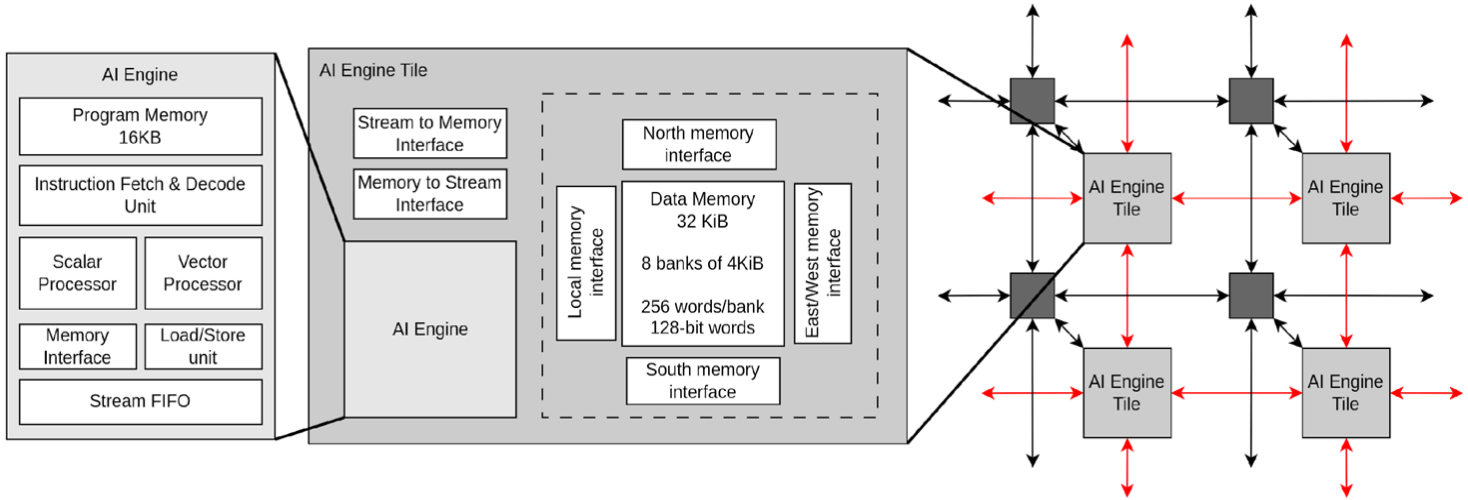
Versal Adaptive SoC



Array of AI Engines

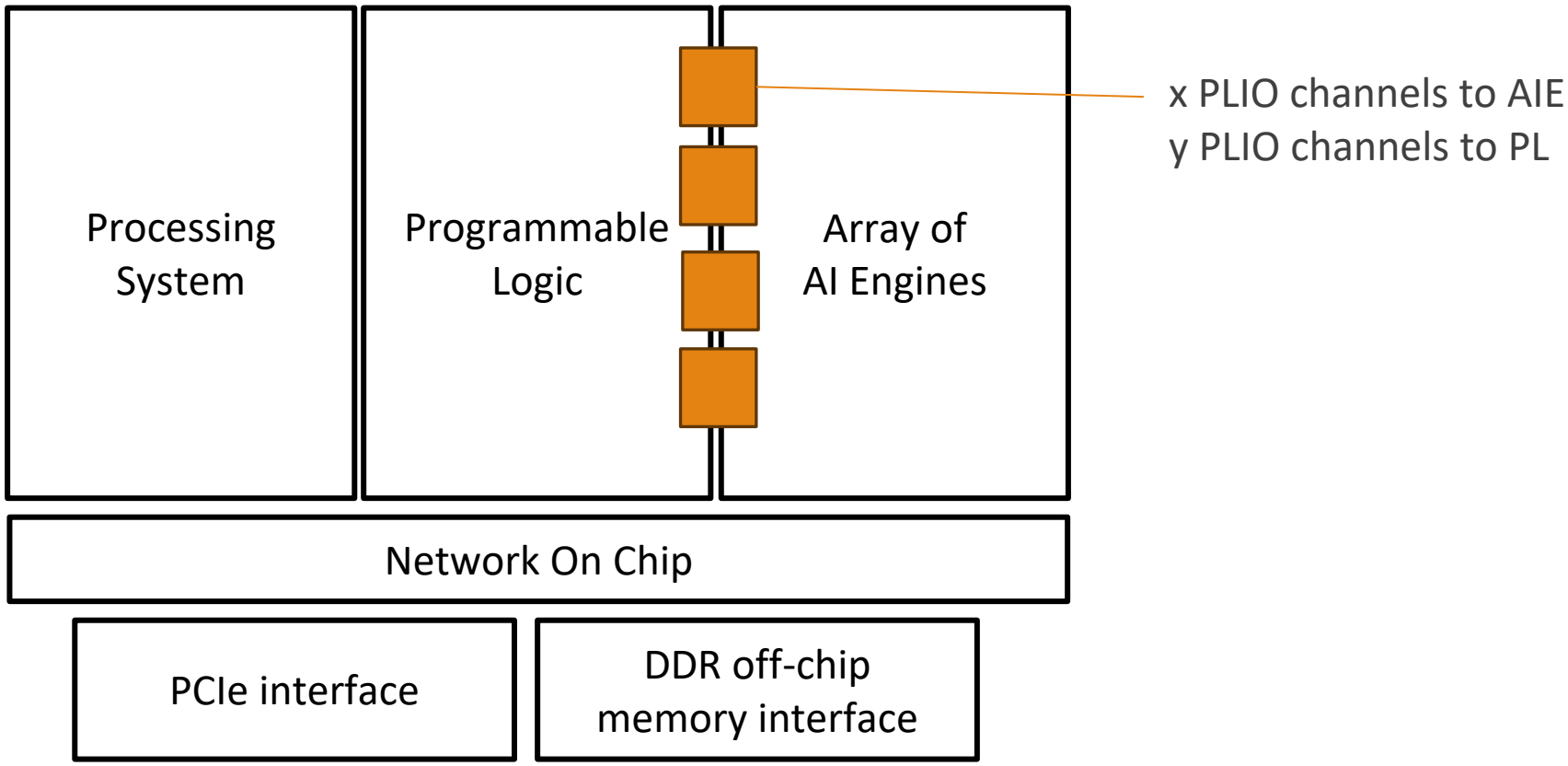
Array
- AI Engine Tiles
- Stream connection
  - 2rd/2wr 32-bit
- Memory connection
  - 2rd 1wr 256-bit

Tile
- AI Engine
- Local Memory



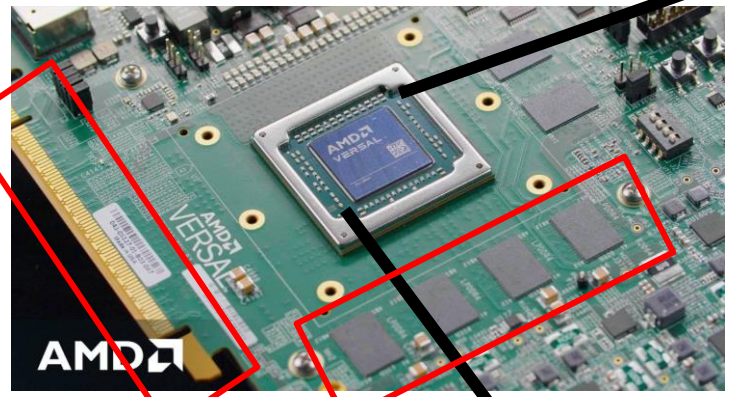
AI Engine
- Very Long Instruction Word (7 instructions per cycle)
- Single Instruction Multiple Data (8 floating point operations per cycle)

Interface tiles



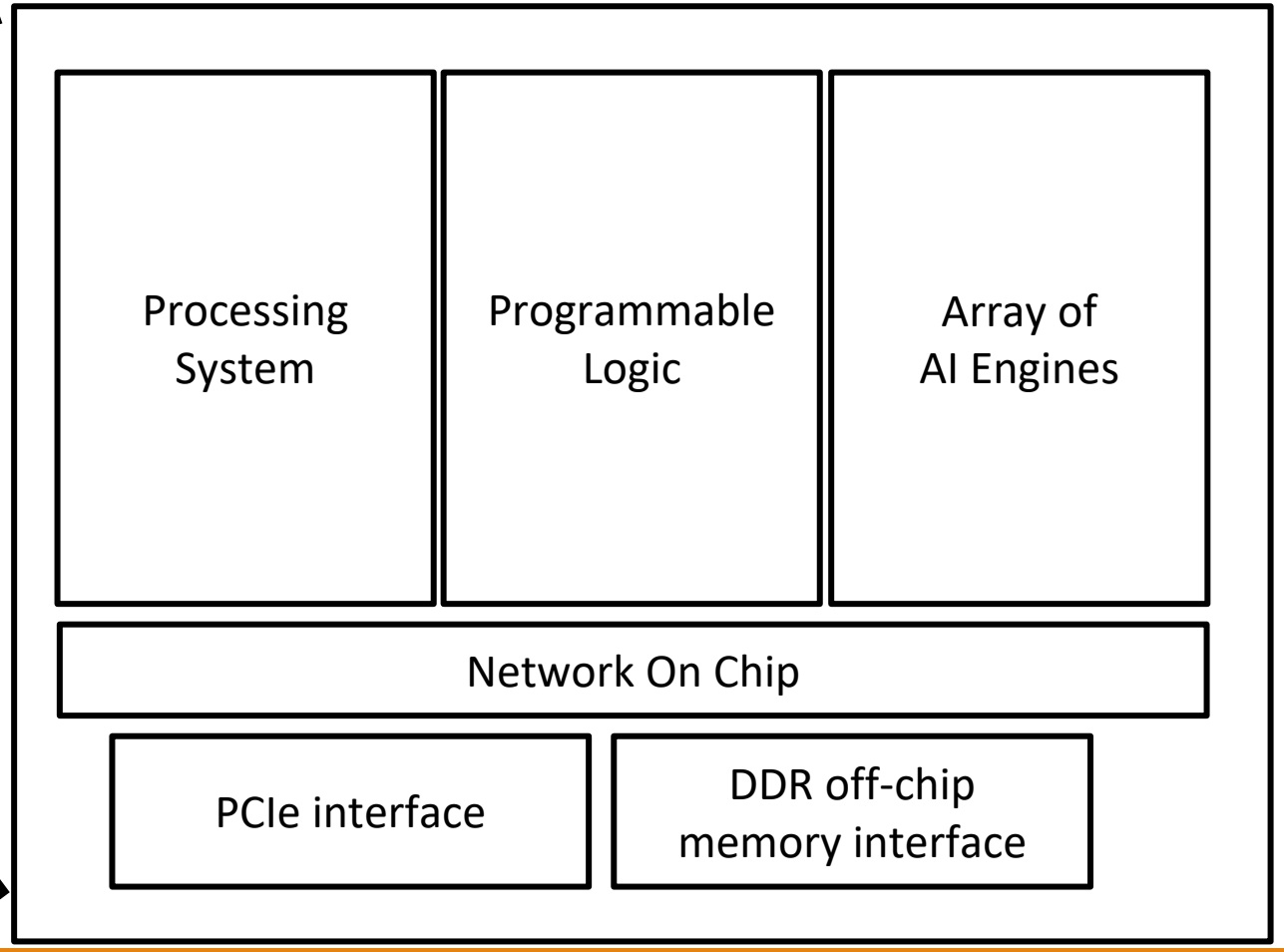
Versal Datacenter Card

PCIe connector



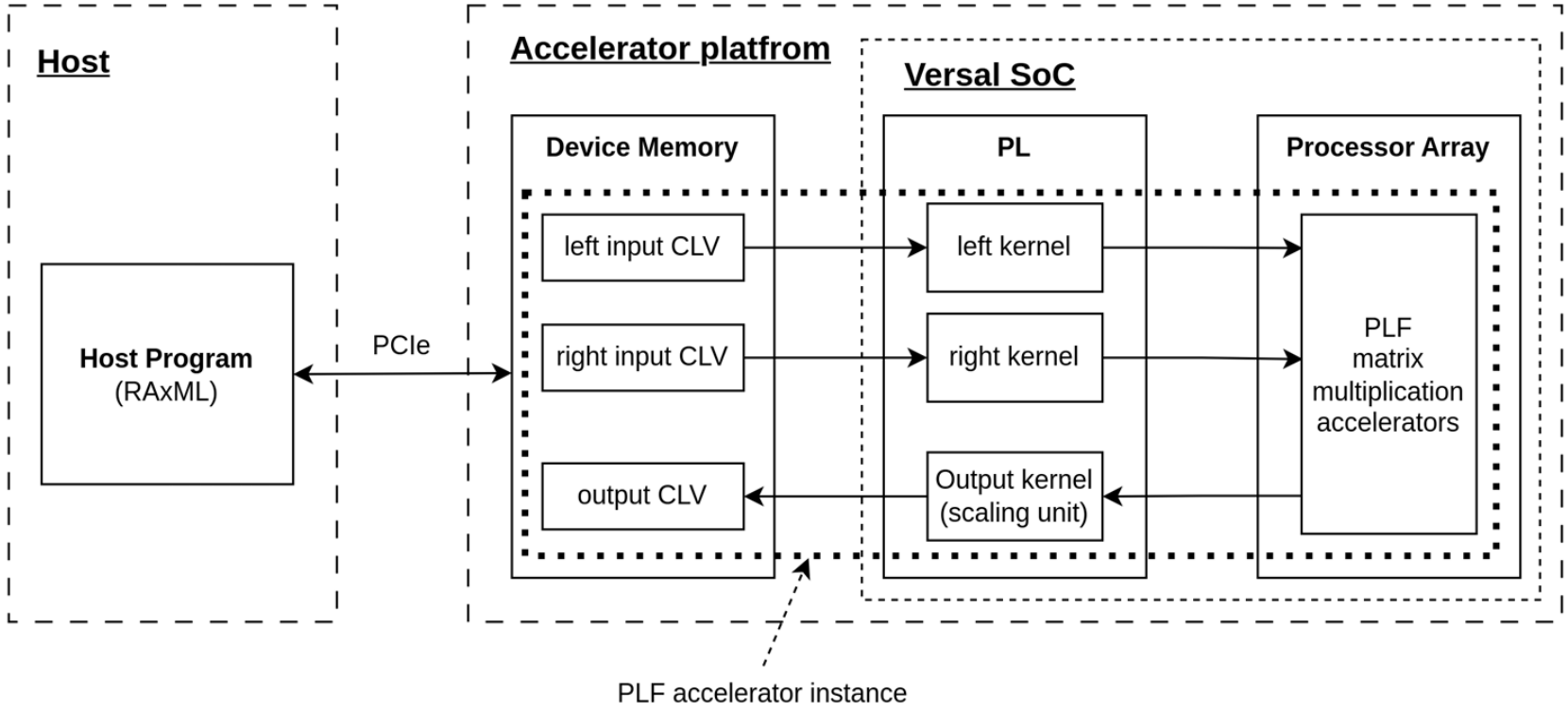
DDR4 memory modules

AMD Versal
VCK5000

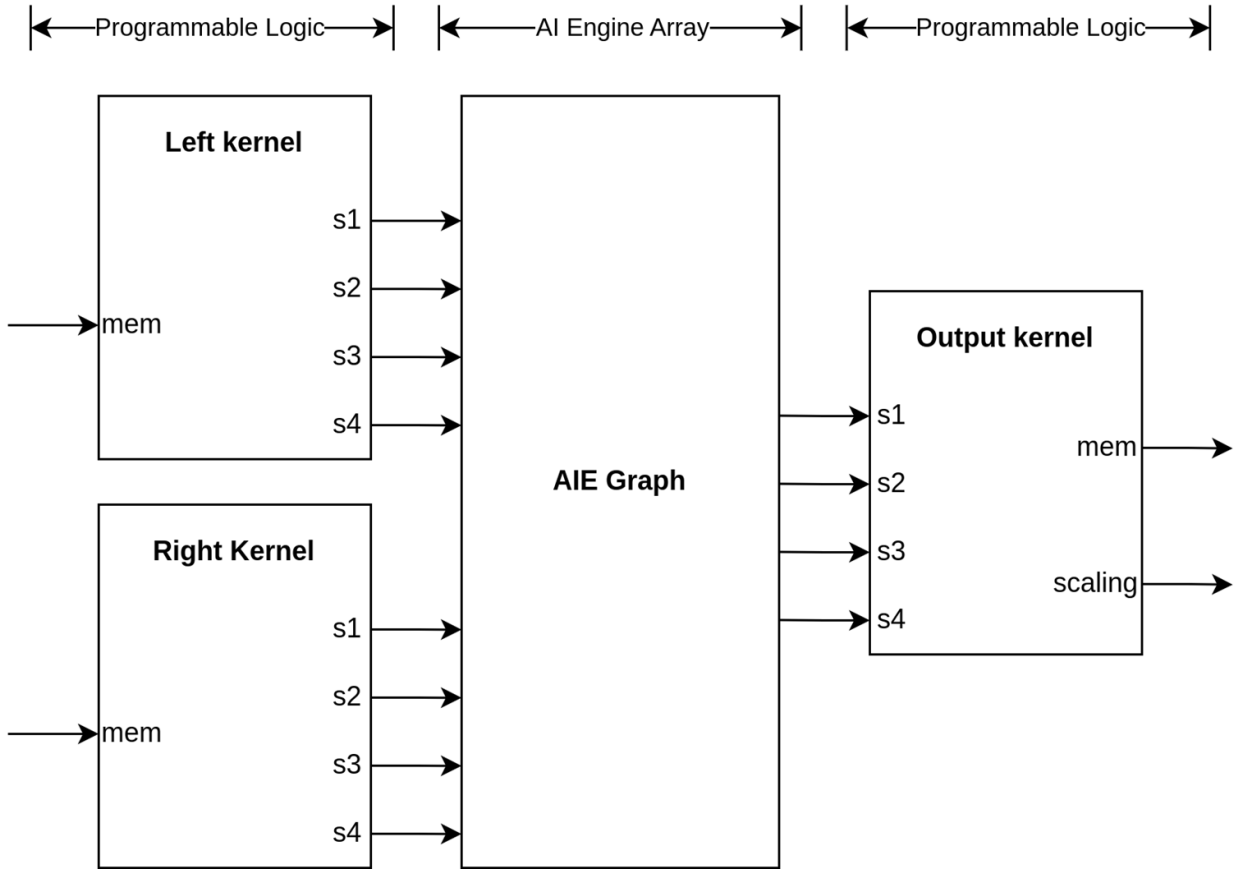


System Architecture

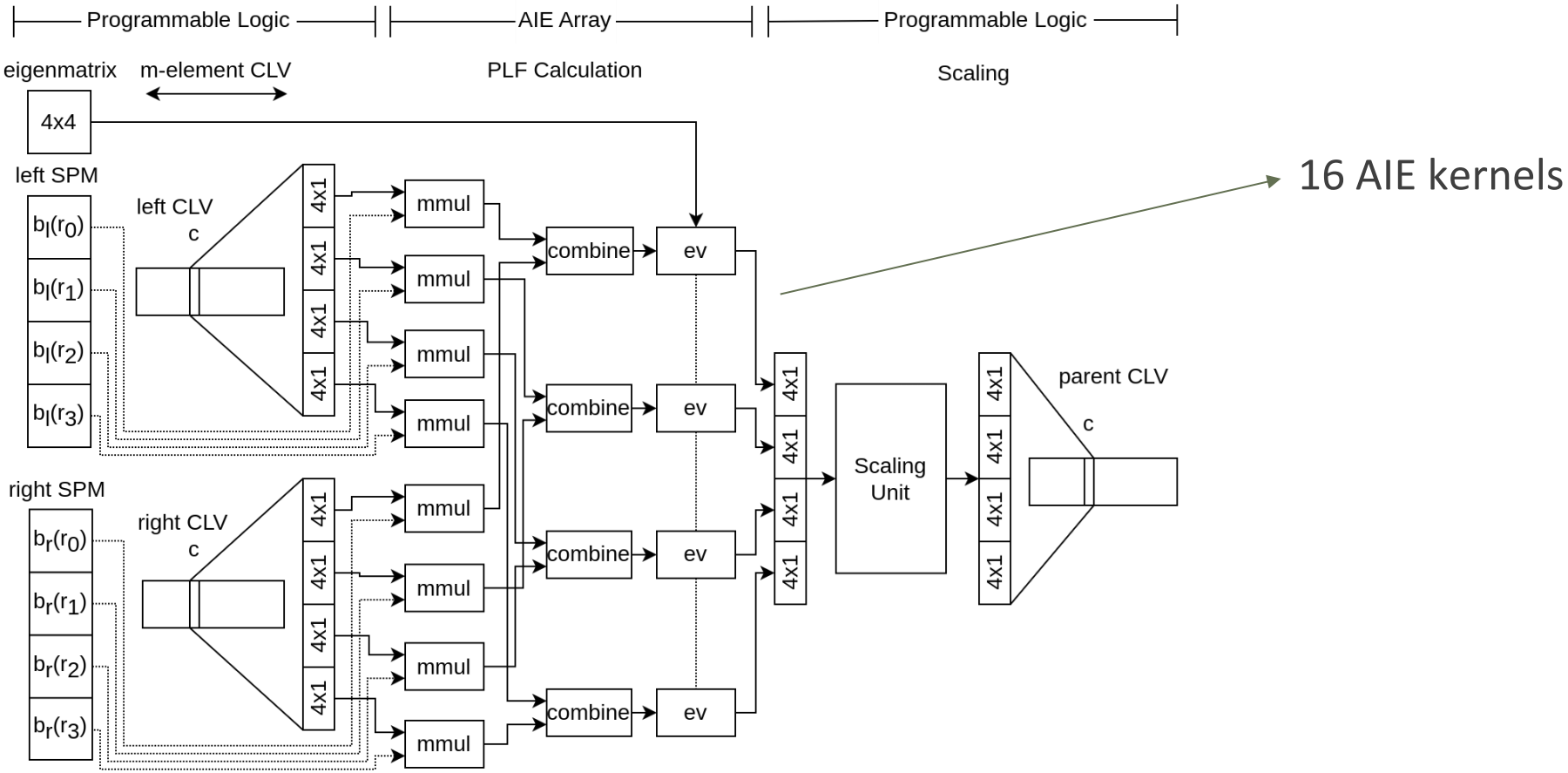
General mapping



PL kernels

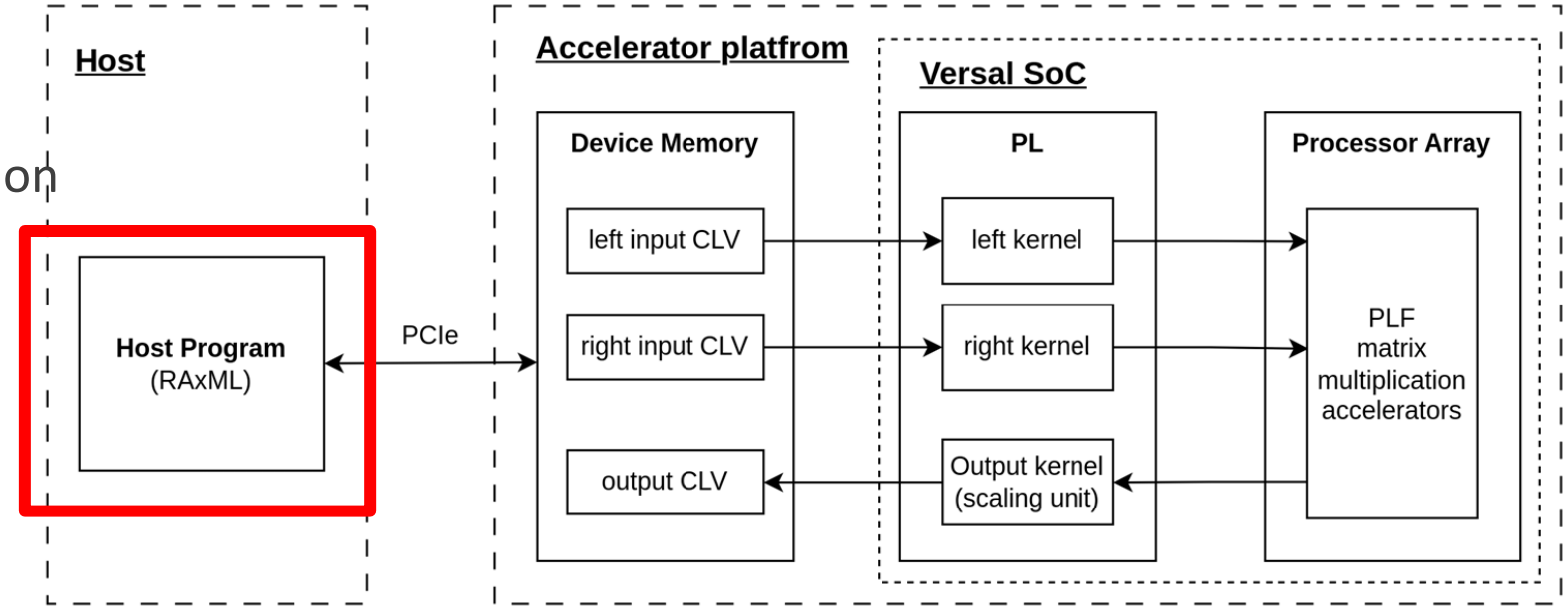

Programmable Logic
AI Engine Array
Programmable Logic
Left kernel
s1
s2
s3
s4
mem
Right Kernel
s1
s2
s3
s4
mem
AIE Graph
Output kernel
s1
s2
s3
s4
mem
scaling

Detailed overview



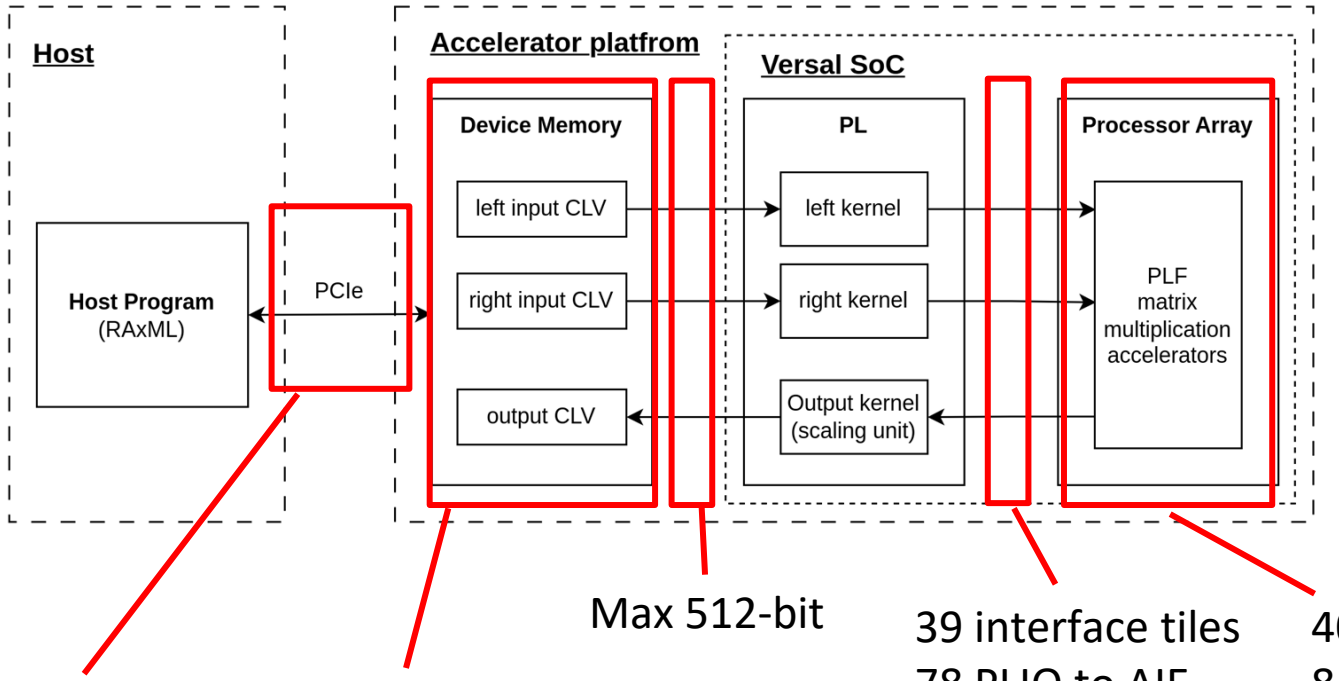
Host program

- Moves data between host and device memory
- Controls platform execution
  - Kernel parameters
  - Start/stop kernels
- Only control over PL kernels



Implementation

Hardware platform



8 lanes PCIe Gen 4.0

4x 4GB DDR4 modules
3 available (12GB)

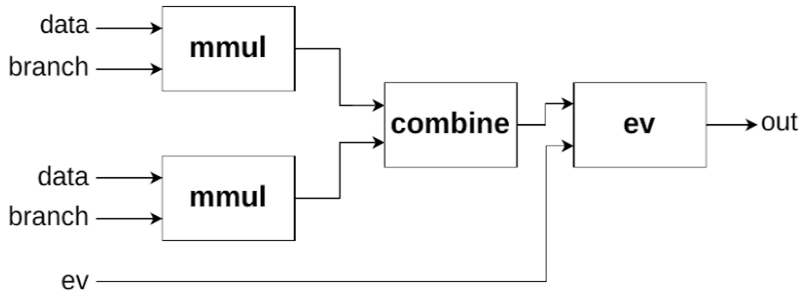
Max 512-bit

39 interface tiles
78 PLIO to AIE
58 PLIO to PL
128-bit channels

400 AIE tiles
8 rows
50 columns

PLIO layouts

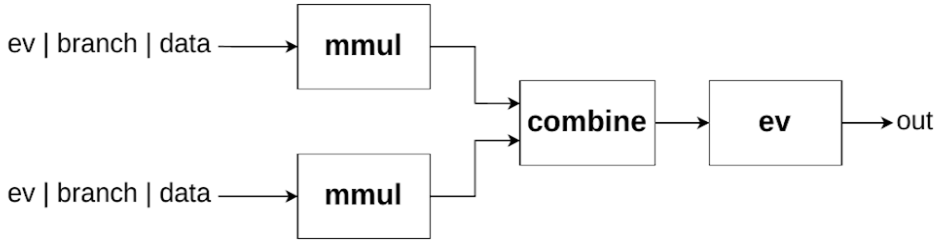
SEPARATE



Each matrix via separate PLIO

- Max 78/17 = 4 instances (87% PLIO utilization)
- 16 tiles x 4 instances = 64 tiles total
- 64/400 = 16% used of AIE array

COMBINED



Matrices share PLIO with left/right CLVs

- Max 78/8 = 9 instances (92% PLIO utilization)
- 16 tiles x 9 instances = 144 tiles total
- 144/400 = 36% used of AIE array

inter-AIE kernel communication

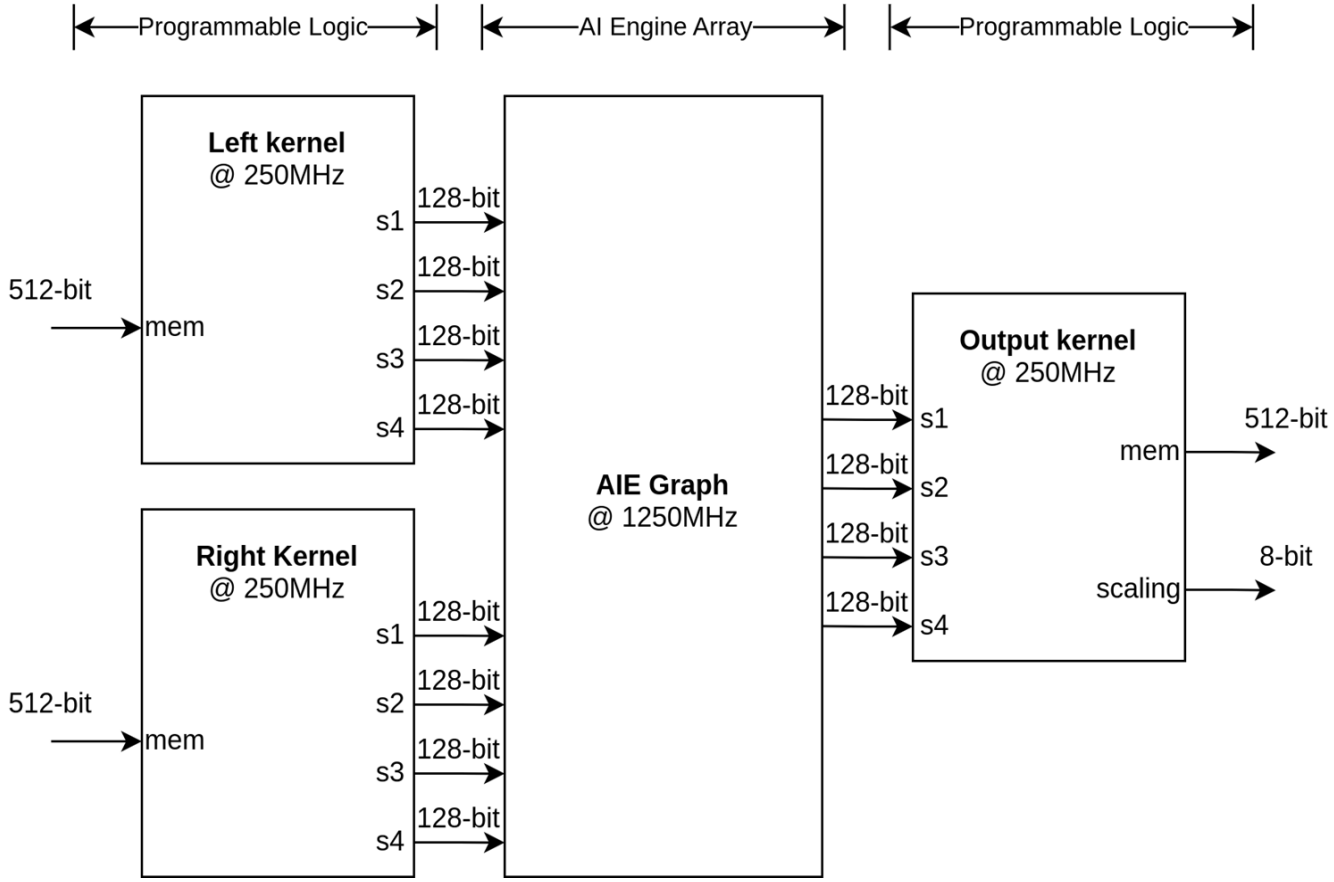
- Stream connections (stream)
  - operates as FIFO of certain length
- Memory connections (window)
  - fixed size memory blocks
    - 1 KiB
    - 8 KiB
    - 16 KiB

AIE tile utilization with windows

	Separate 4 instances	Combined 9 instances
compute only	16%	36%
1-KiB window	23%	41%
8-KiB window	25%	62%
16-KiB window	34%	76%

Programmable Logic utilization

- Limited to 250MHz
- <4.5% of the PL resources per instance
- for 9 instances:
  - < 41% of PL resources used



Evaluation

Experimental Setup

- Heterogeneous Accelerated Compute Cluster (HACC) at ETH Zürich
  - 2 AMD EPYC CPUs (total 128 cores)
  - 2 AMD Versal VCK5000 cards (1 used)

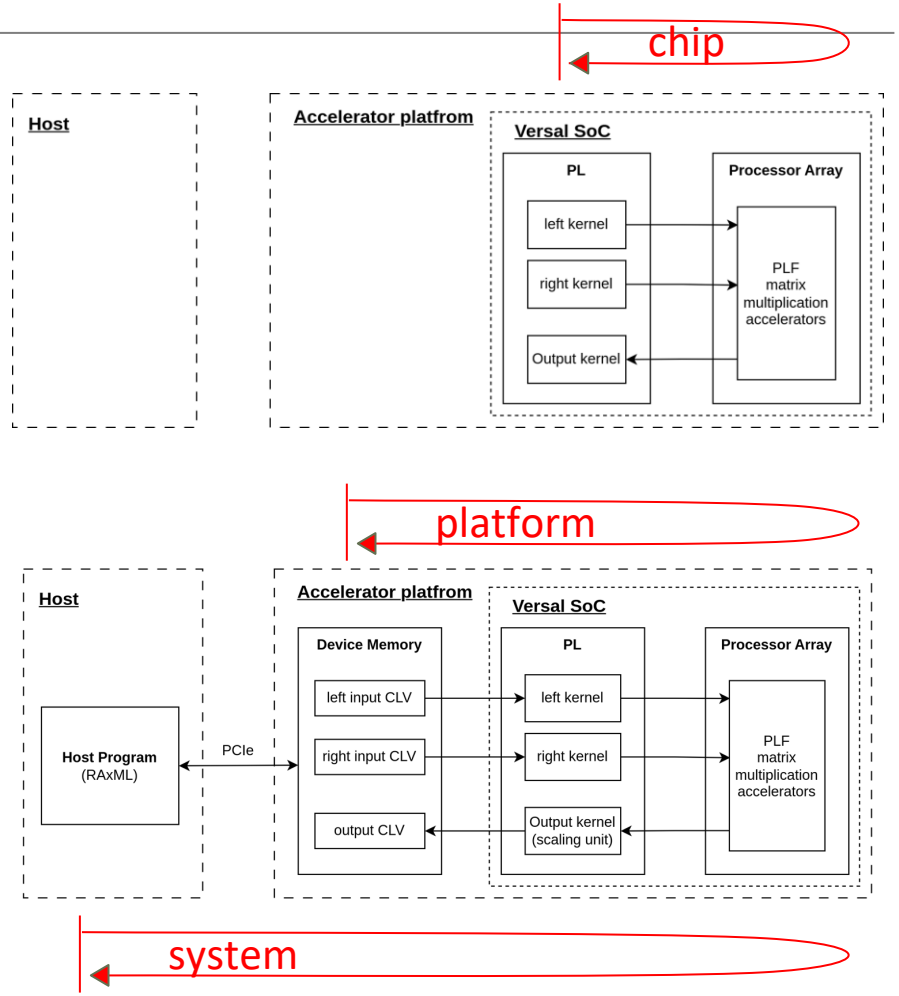
- design exploration variables
  - number of instances: 1,2,4,8 and 9 instances
  - CLV lengths: 100 to 10M sites
  - AIE communication methods: Stream, window (1, 8, 16 KiB)
  - PLIO layout: Separate, combined

- Evaluation metric
  - Throughput in CLV entries per second (CLVES)
  - 1 CLV entry = 16 single precision floating point values

Design Space Exploration

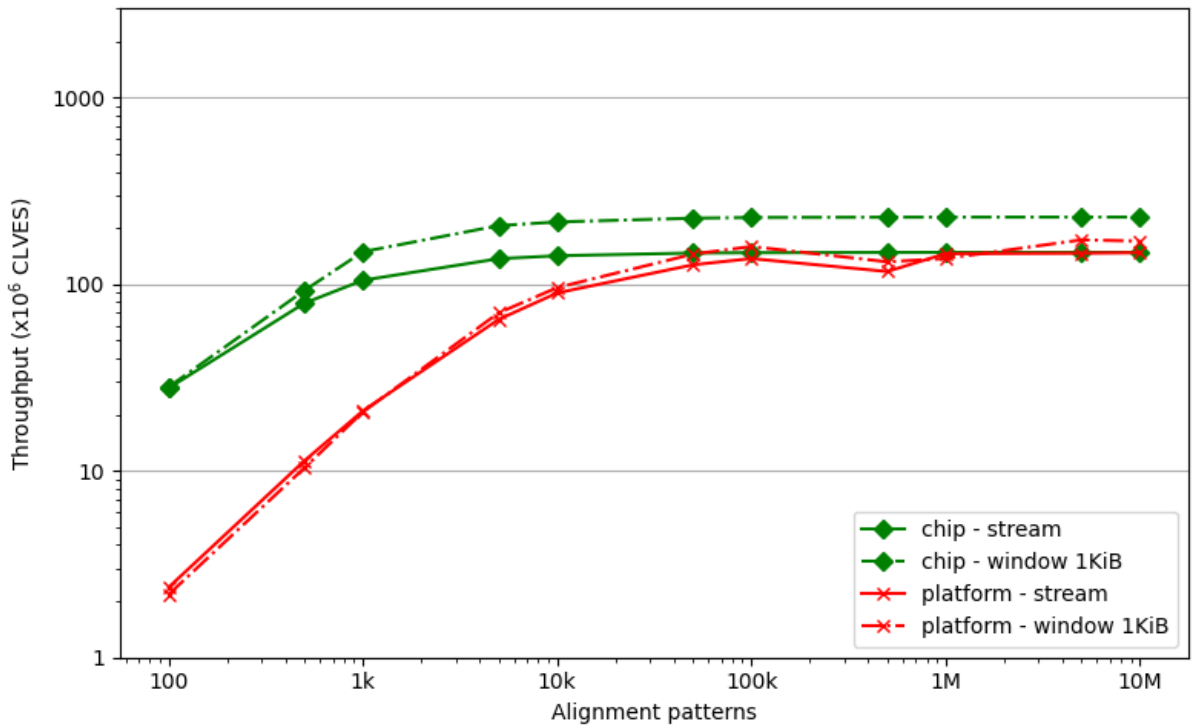
1) Chip performance
   - Custom PL kernels generating test data
   - PLF on AIE
   - Measure PL-AIE throughput

2) Platform/system performance
   - complete functionality
   - Measure platform throughput (excludes pcie transfers)
   - Measure system throughput (includes pcie transfers)



Stream vs Window (Separate Layout)

- Stream:
  - operates as FIFO of certain length
  - only needs matrices once
- Window:
  - fixed size memory blocks
  - needs to resend matrices for each block

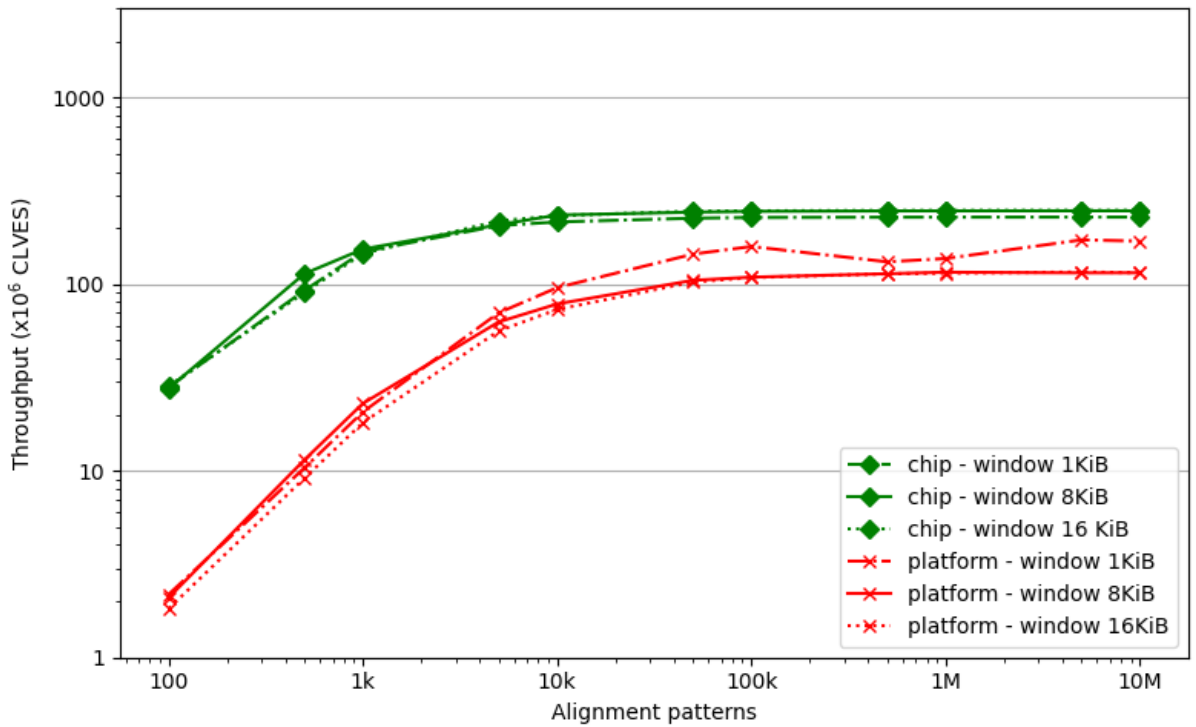


Take away:

Window-based communication outperforms streams

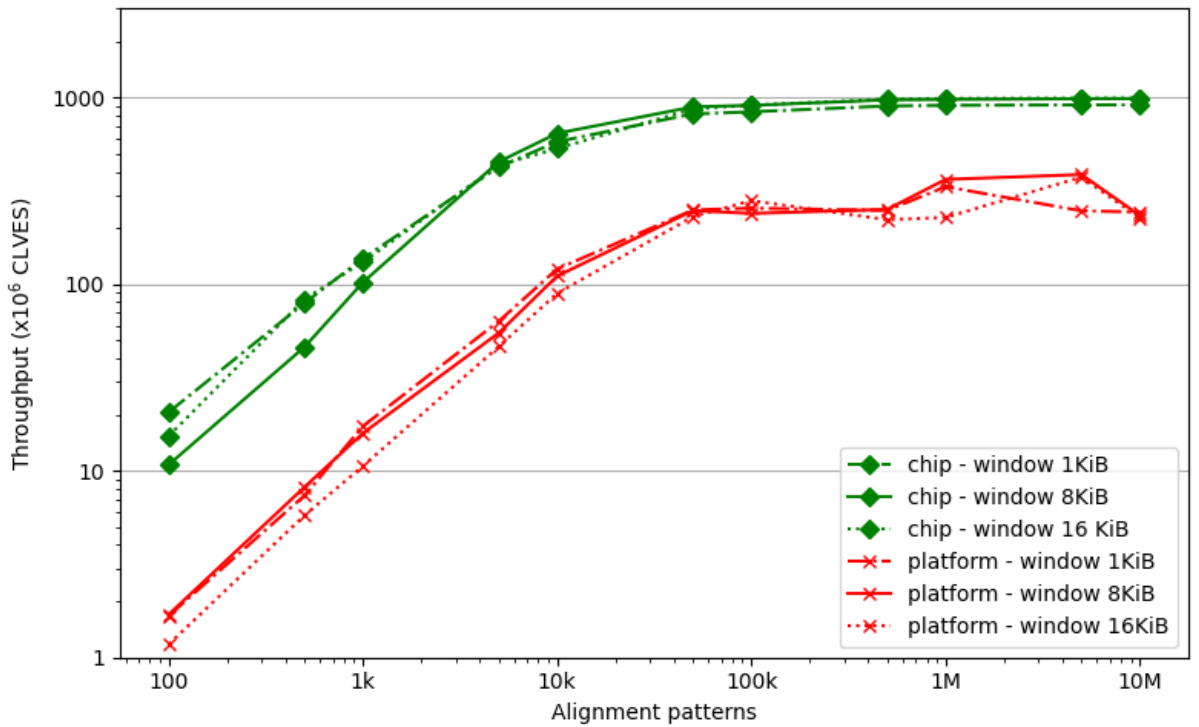
Varying window sizes (Separate layout)

- window sizes:
  - 1 KiB
  - 8 KiB
  - 16 KiB



Varying window sizes (Separate layout, 4 instances)

- window sizes:
  - 1 KiB
  - 8 KiB
  - 16 KiB



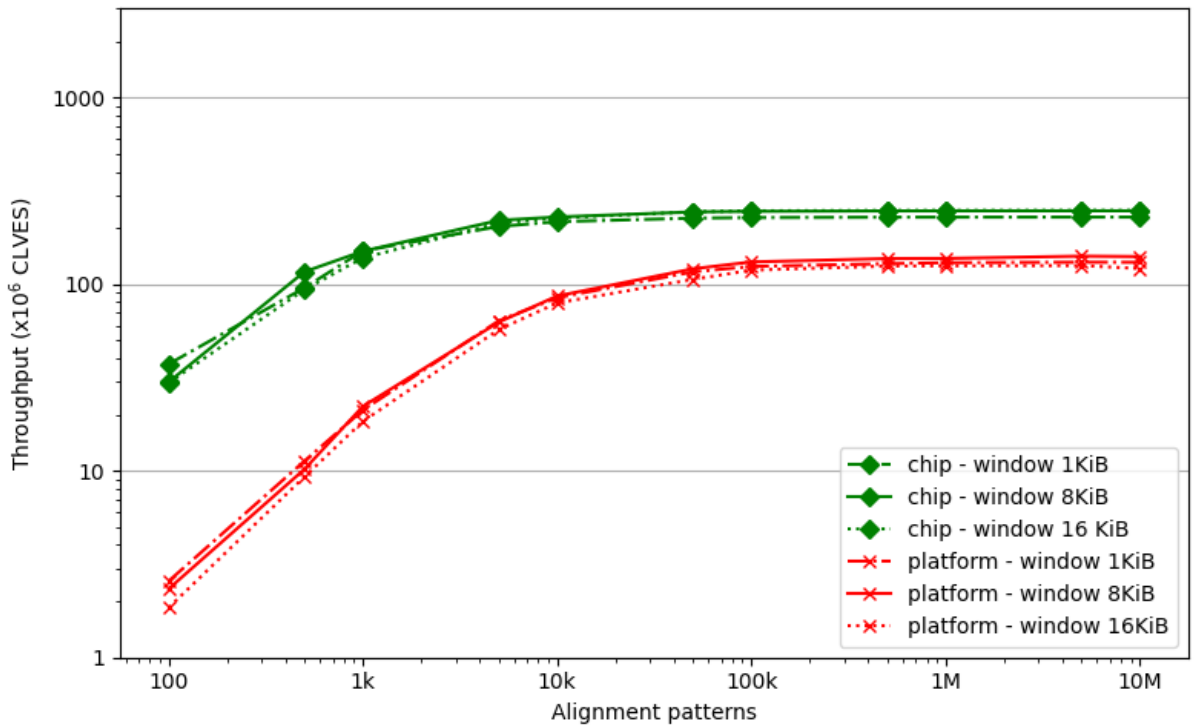
Take away:

marginal difference window sizes

1 KiB and 8 KiB seem fastest for Separate layout

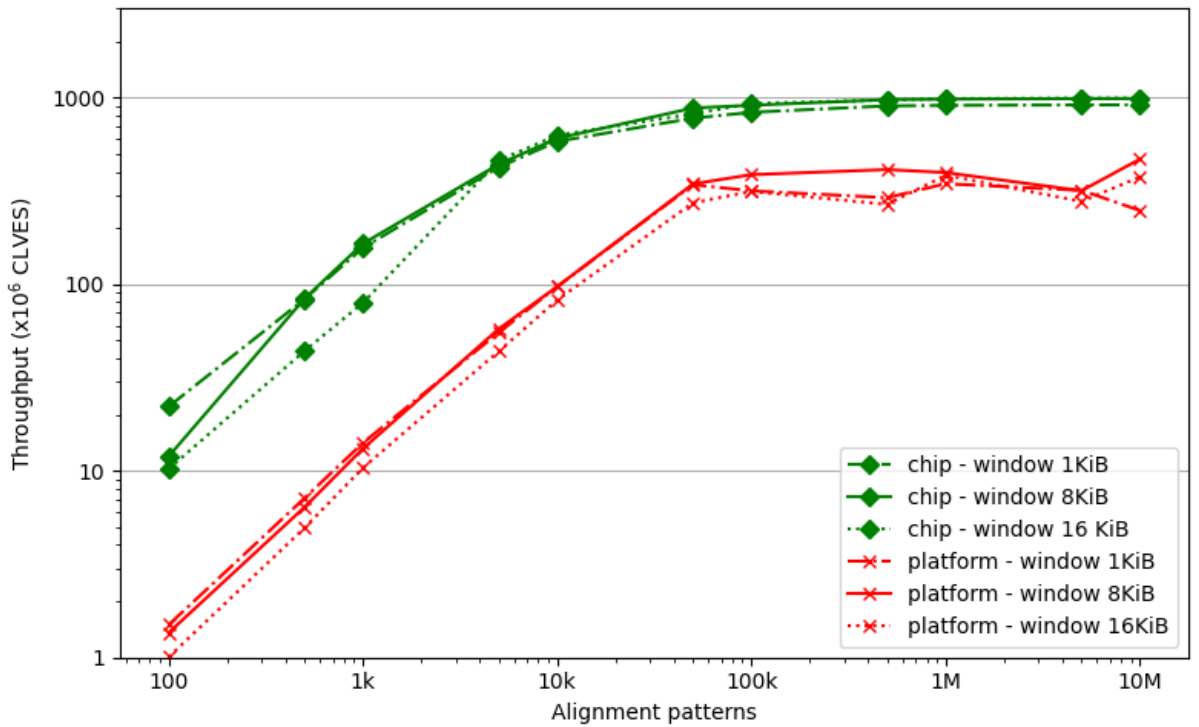
Varying window sizes (Combined layout, 1 instance)

- window sizes:
  - 1 KiB
  - 8 KiB
  - 16 KiB



Varying window sizes (Combined layout, 4 instances)

- window sizes:
  - 1 KiB
  - 8 KiB
  - 16 KiB



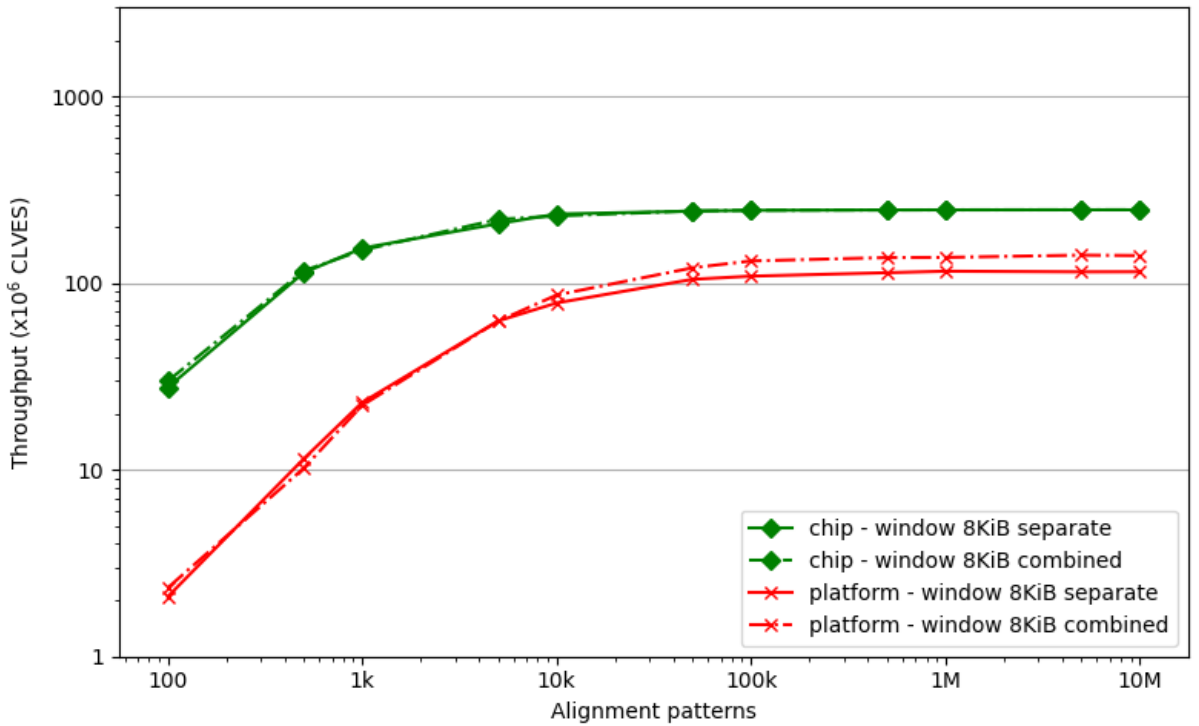
Take away:

marginal difference window sizes

8KiB seems generally fastest for the Combined layout

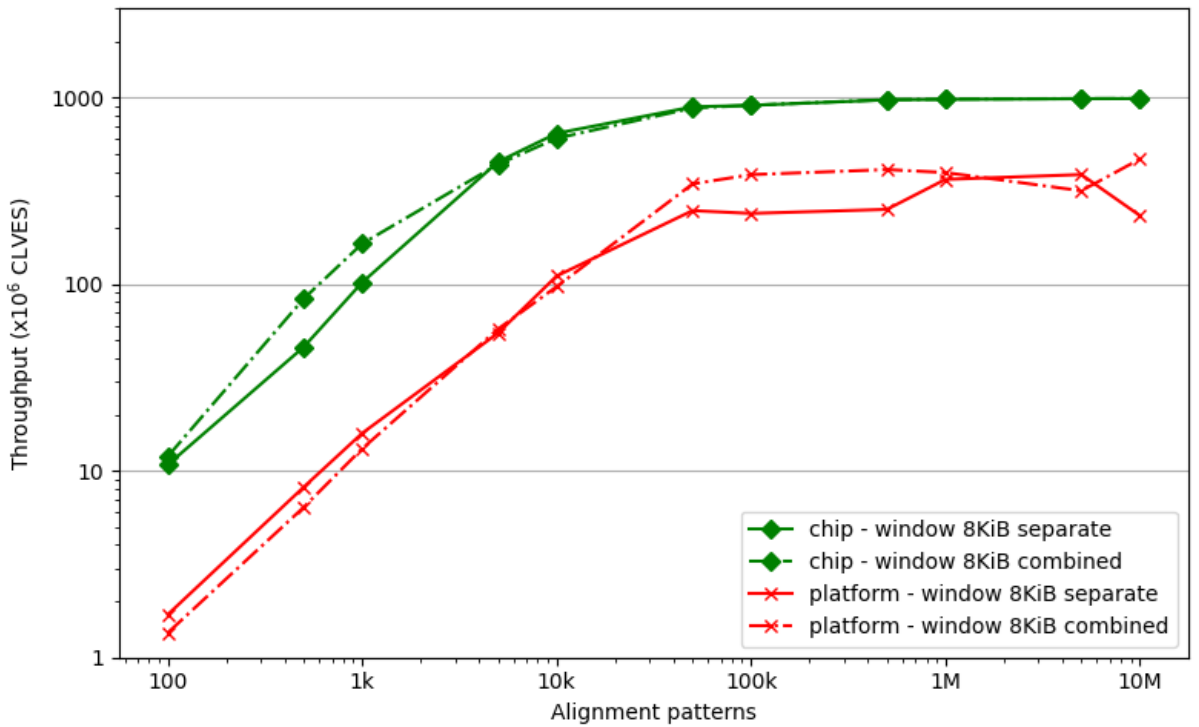
Separate vs Combined layout (8KiB window, 1instance)

- Separate Layout:
  - each matrix send over individual PLIO channel
  - max 4 instances limited by PLIO
- Combined Layout:
  - Matrices share PLIO channels with CLV data
  - max 9 instances limited by PLIO



Separate vs Combined layout (8KiB window, 4 instances)

- Separate Layout:
  - each matrix send over individual PLIO channel
  - max 4 instances limited by PLIO
- Combined Layout:
  - Matrices share PLIO channels with CLV data
  - max 9 instances limited by PLIO



Take away:

marginal difference between layouts

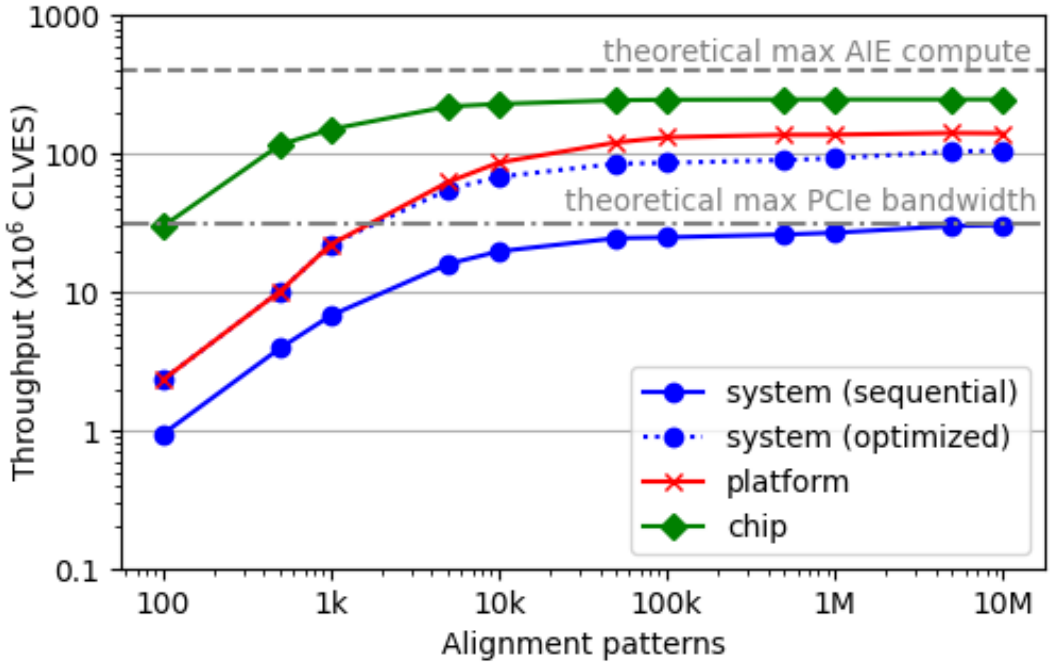
Combined seems slightly faster

Combined preferred over Separate

System performance (8KiB window, Combined, 1 instance)

- Fastest configuration:
  - window based
  - 8 KiB window
  - Combined PLIO

- PCIe optimizations:
  - Double buffering
  - caching
  - performance model: 3.45x speedup



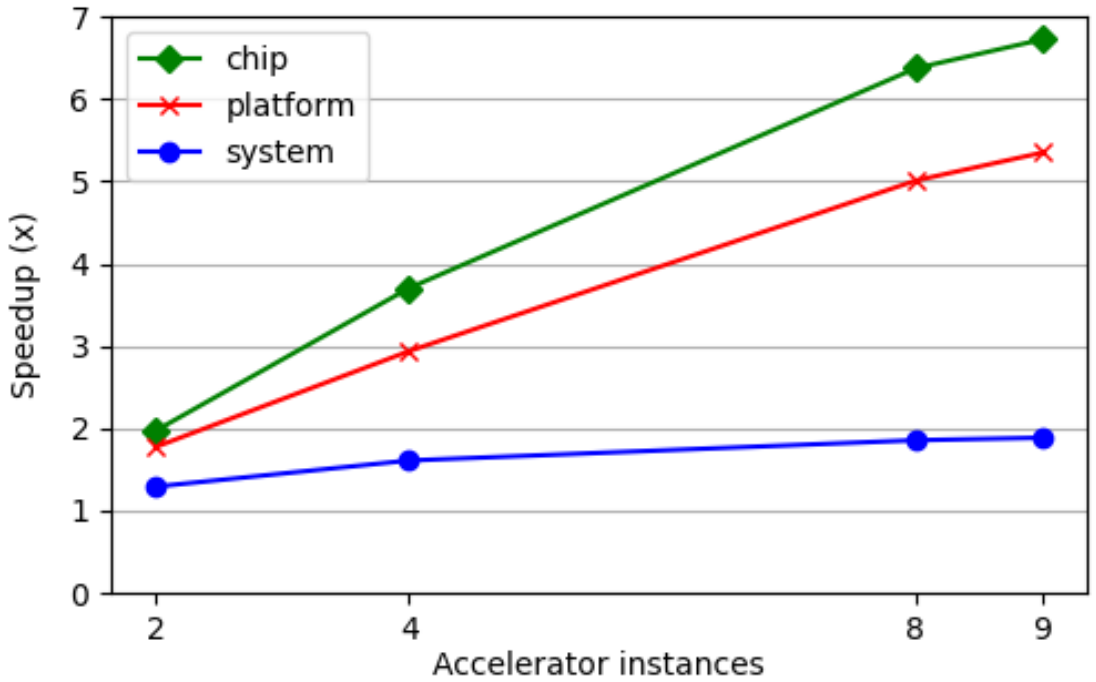
Take away:

PCIe bandwidth limits system performance

Optimizations may improve system performance

Scalability

- Chip:
  - PL - AIE bandwidth
- Platform:
  - includes device memory access
- System:
  - includes host-device transfers over PCIe



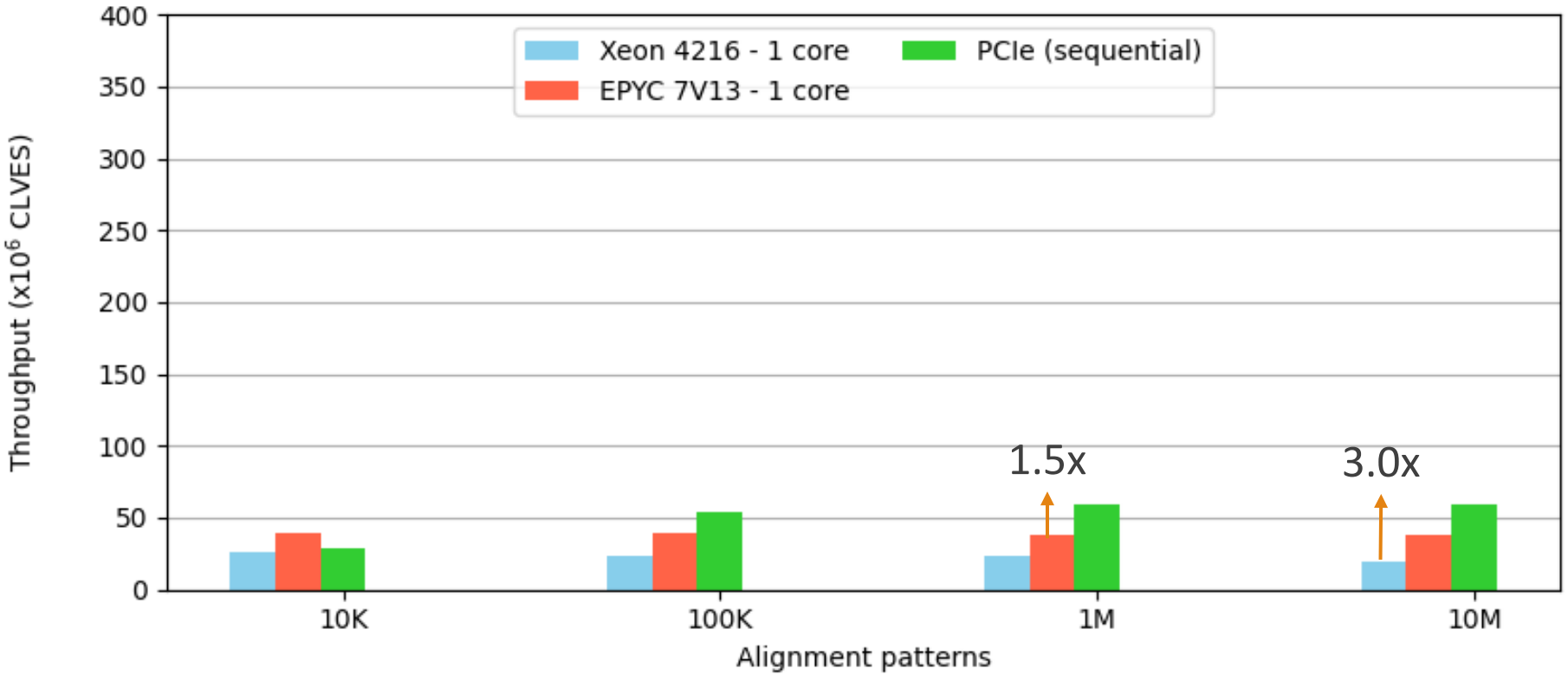
Take away:

chip/platform scale linearly with increasing instances (almost 1:1)

System can't scale due to PCIe bandwidth limit

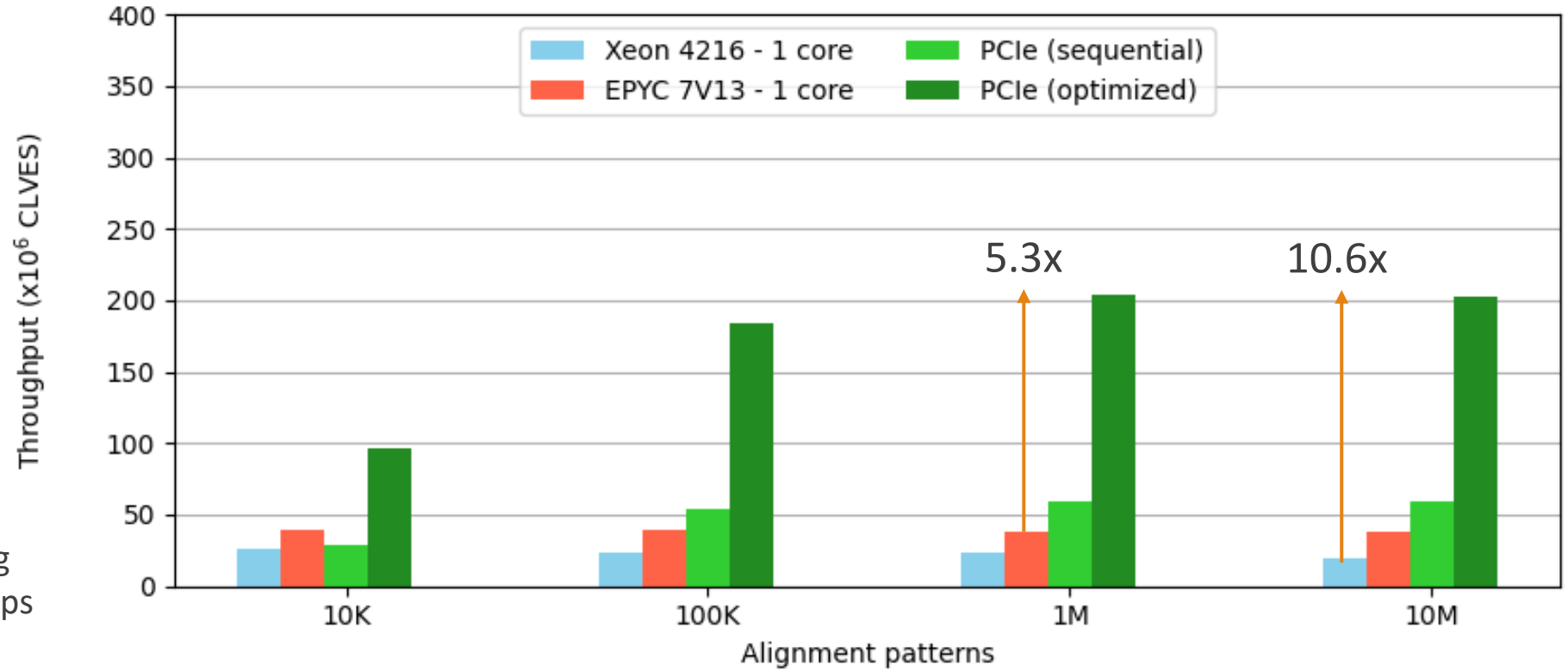
CPU comparison
1 core vs system (sequential)

- High-end server CPUs
  - AVX2 vector extensions
  - highly optimized CPU implementation of PLF
- system sequential:
  - no reuse of data
  - no overlap between movement and computation



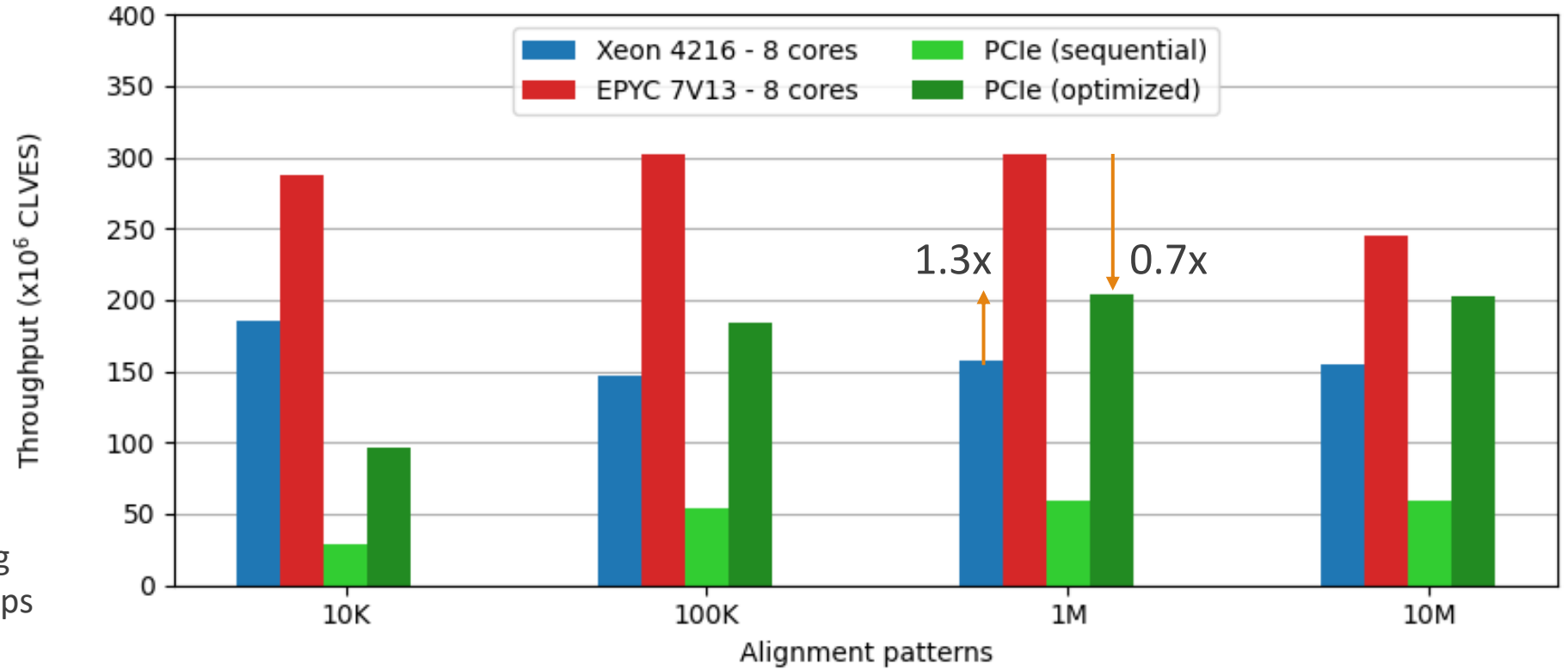
CPU comparison
1 core vs system (optimized)

- High-end server CPUs
  - AVX2 vector extensions
  - highly optimized CPU implementation of PLF
- system sequential:
  - no reuse of data
  - no overlap between movement and computation
- system optimized:
  - data reuse from caching
  - double buffering overlaps movement and computation



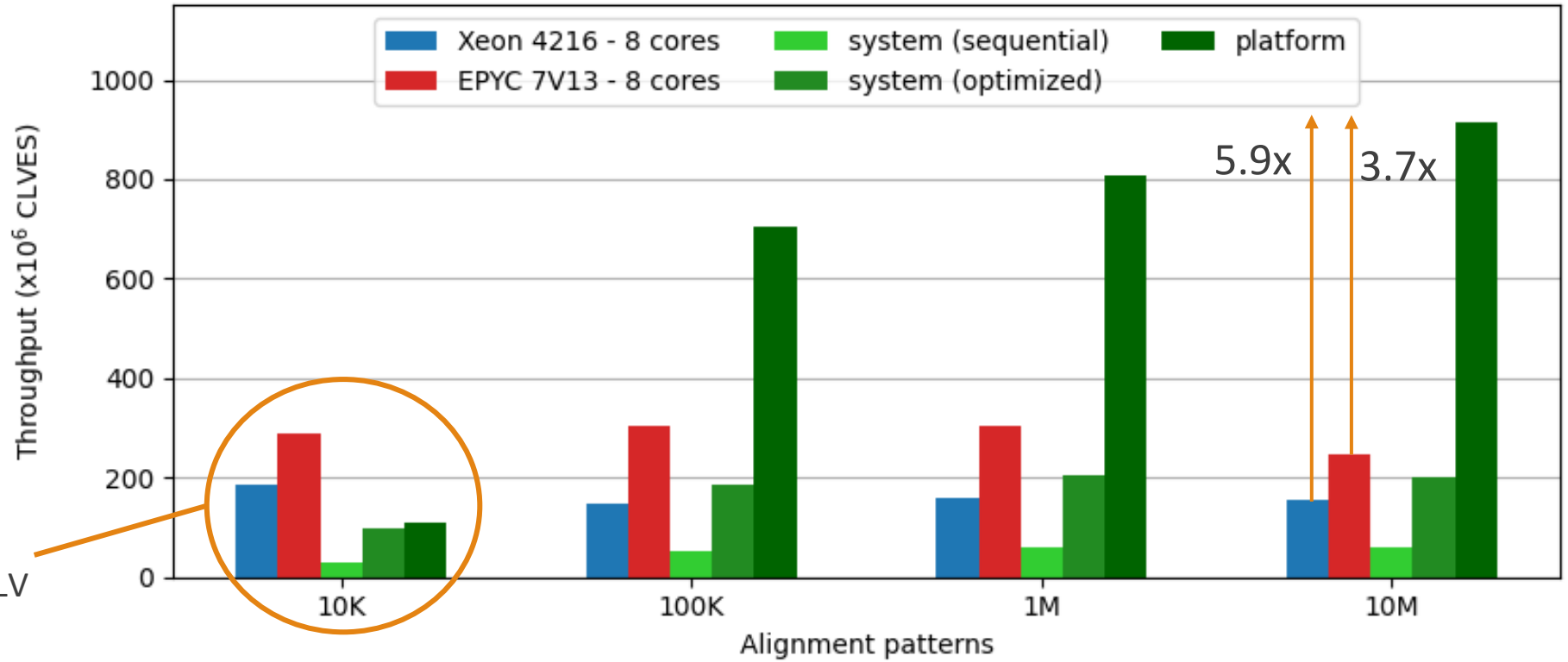
CPU comparison
8 cores vs system (optimized)

- High-end server CPUs
  - AVX2 vector extensions
  - highly optimized CPU implementation of PLF
- system sequential:
  - no reuse of data
  - no overlap between movement and computation
- system optimized:
  - data reuse from caching
  - double buffering overlaps movement and computation



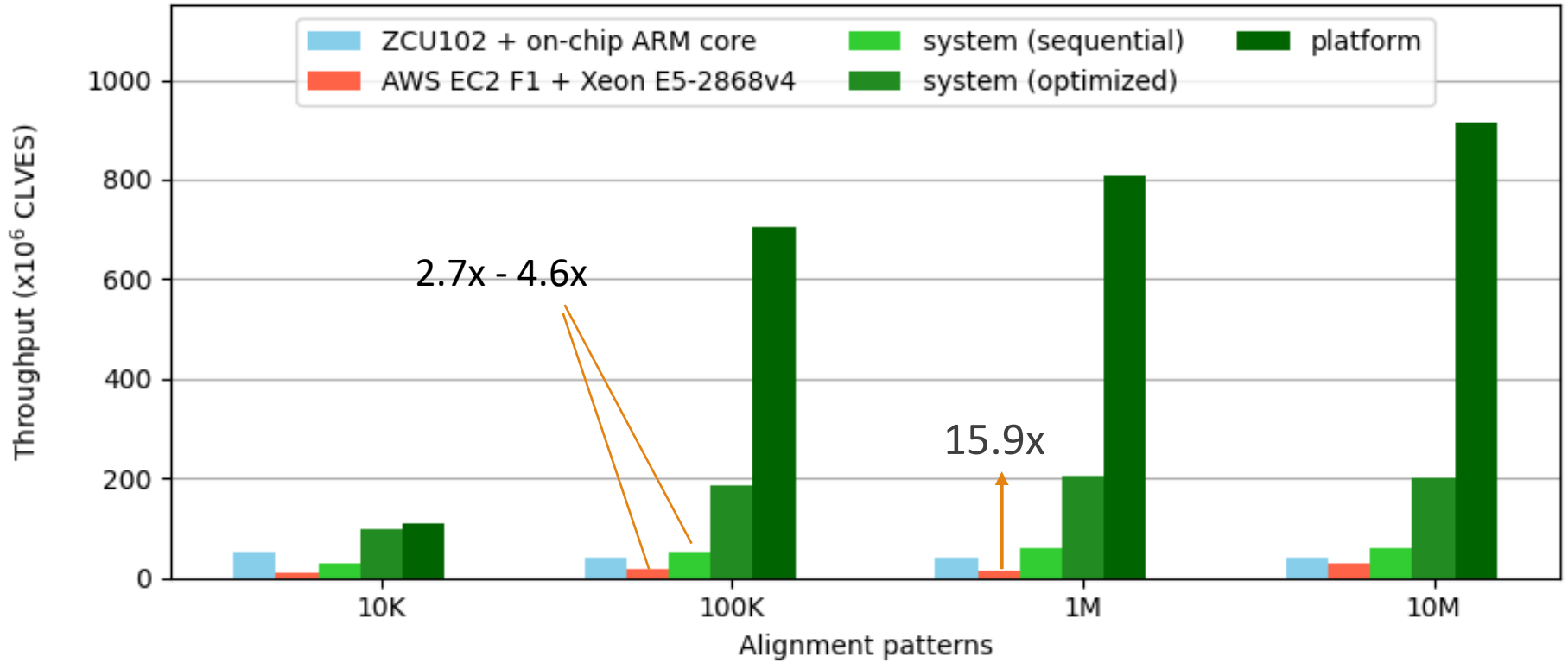
CPU comparison
8 cores vs platform

- High-end server CPUs
  - AVX2 vector extensions
  - highly optimized CPU implementation of PLF
- platform
  - no pcie transfer
  - indication of performance when host program on on-chip ARM CPU
- impractical use of hw acceleration for short CLV lengths



FPGA comparison
System 1 - AWS EC2 F1

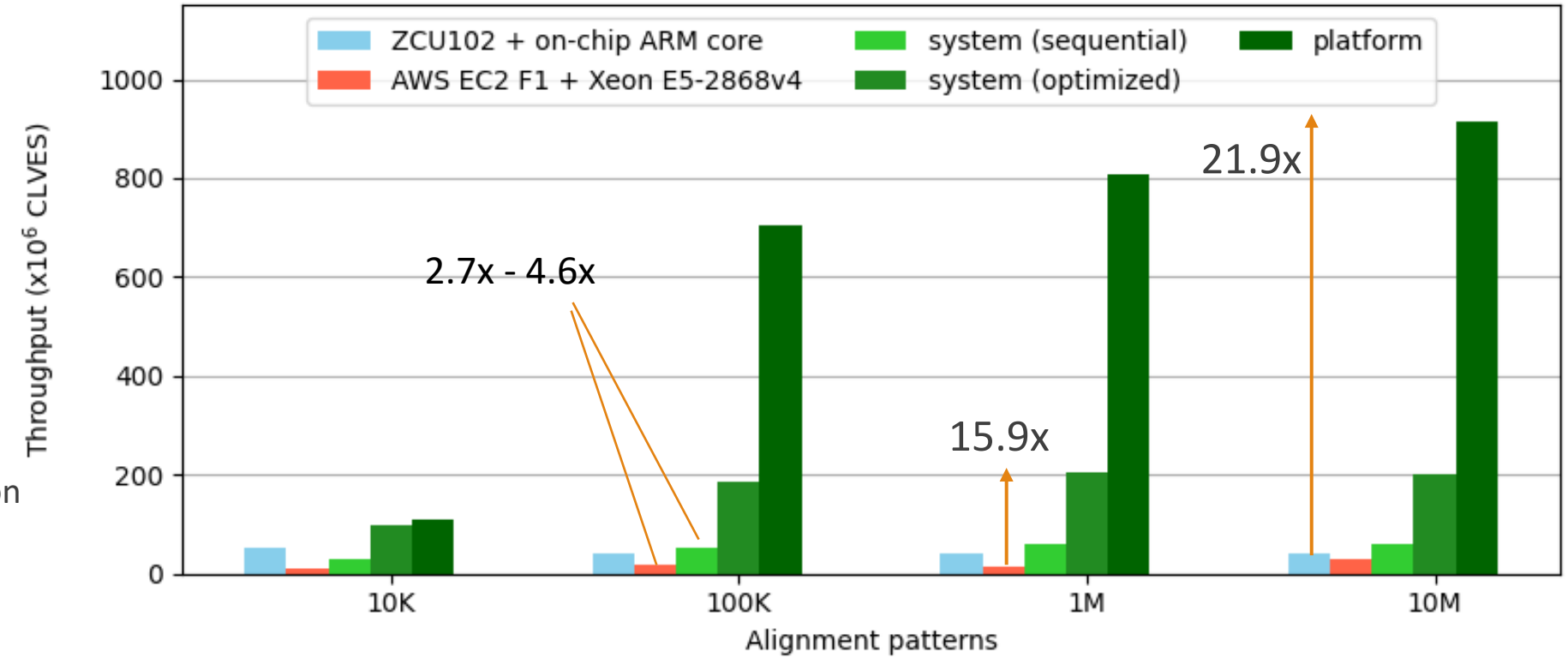
- AWS EC2 F1
  - Cloud-based FPGA
  - moves data between host and device over PCIe
  - Uses optimizations



Malakonakis, P., Brokalakis, A., Alachiotis, N., Sotiriades, E., & Dollas, A. (2020, October). Exploring modern FPGA platforms for faster phylogeny reconstruction with RAxML. In *2020 IEEE 20th International Conference on Bioinformatics and Bioengineering (BIBE)* (pp. 97-104). IEEE.

FPGA comparison
System 2 - ZCU102

- AWS EC2 F1
  - Cloud-based FPGA
  - moves data between host and device over PCIe
  - Uses optimizations
- ZCU102
  - Development board
  - Host and device are on same SoC and share memory
  - no PCIe transfers needed



Malakonakis, P., Brokalakis, A., Alachiotis, N., Sotiriades, E., & Dollas, A. (2020, October). Exploring modern FPGA platforms for faster phylogeny reconstruction with RAxML. In *2020 IEEE 20th International Conference on Bioinformatics and Bioengineering (BIBE)* (pp. 97-104). IEEE.

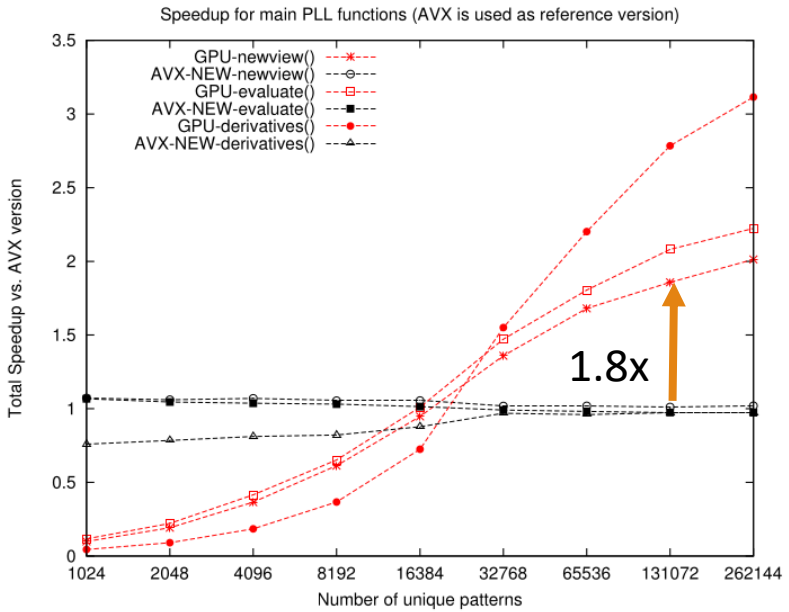
GPU Performance

No direct comparison possible

- Up to 1.8x speedup for 100K
- Compared to one Intel i5-3550 core with AVX intrinsics

Our implementation

- Up to 8.9x speedup for 100K
- Compared to one Intel Xeon Silver 4216 core with AVX2 intrinsics



Izquierdo-Carrasco, F., Alachiotis, N., Berger, S., Flouri, T., Pissis, S. P., & Stamatakis, A. (2013, May). A generic vectorization scheme and a GPU kernel for the phylogenetic likelihood library. In *2013 IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum* (pp. 530-538). IEEE.

Conclusion

Conclusion

- We presented an accelerator architecture for the Phylogenetic Likelihood Function informed by a design space exploration of the AMD Versal Adaptive SoCs

- Design-space exploration takeaways:
  - windows preferred over streams
  - Use PLIO channels sparingly
  - System limited by PCIe data movement
  - Hardware acceleration impractical for CLV lengths < 50K
- Achieved performance
  - Our design vs single high-end CPU core + AVX2: 1.5 - 3x (potentially up to 10x)
  - Our design vs eight high-end CPU cores + AVX2: similar performance
  - Our design vs modern FPGA + host CPU over PCIe: up to 4.6x (potentially up to 16x)
  - Our design vs modern FPGA + integrated CPU: potentially up to 22x

Future work

- implement Protein-based implementation (5x higher arithmetic intensity than DNA)
- Redesign Programmable logic kernels (Monolithic design, explorer 100 Gbit ethernet ports)
- Porting RAxML to Versal ARM cores (eliminating PCIe transfers)

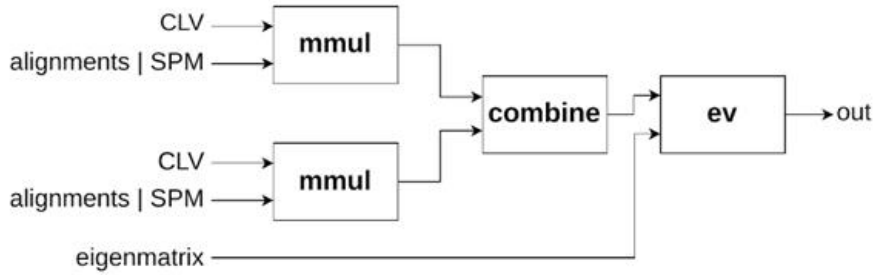
Questions

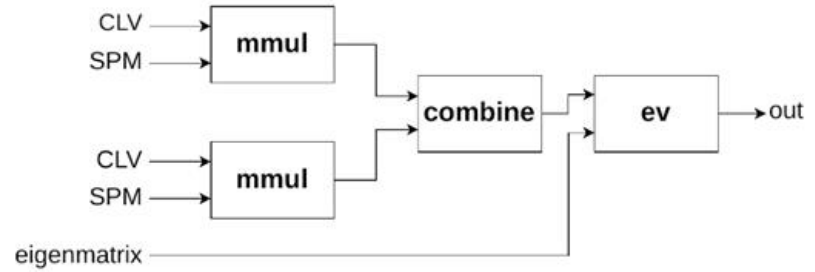
AIE configurations

Stream

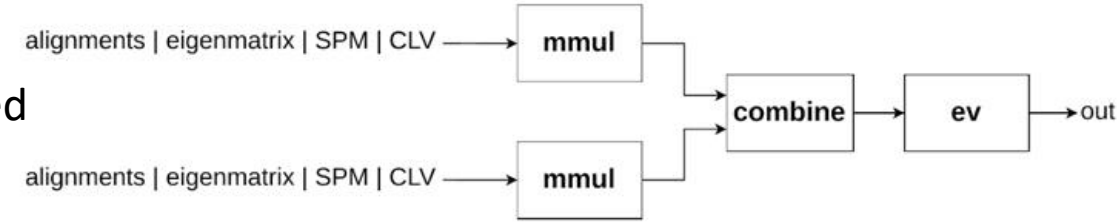
Window

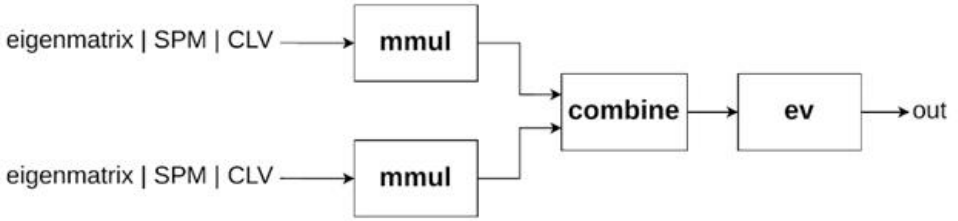
Separate





Combined





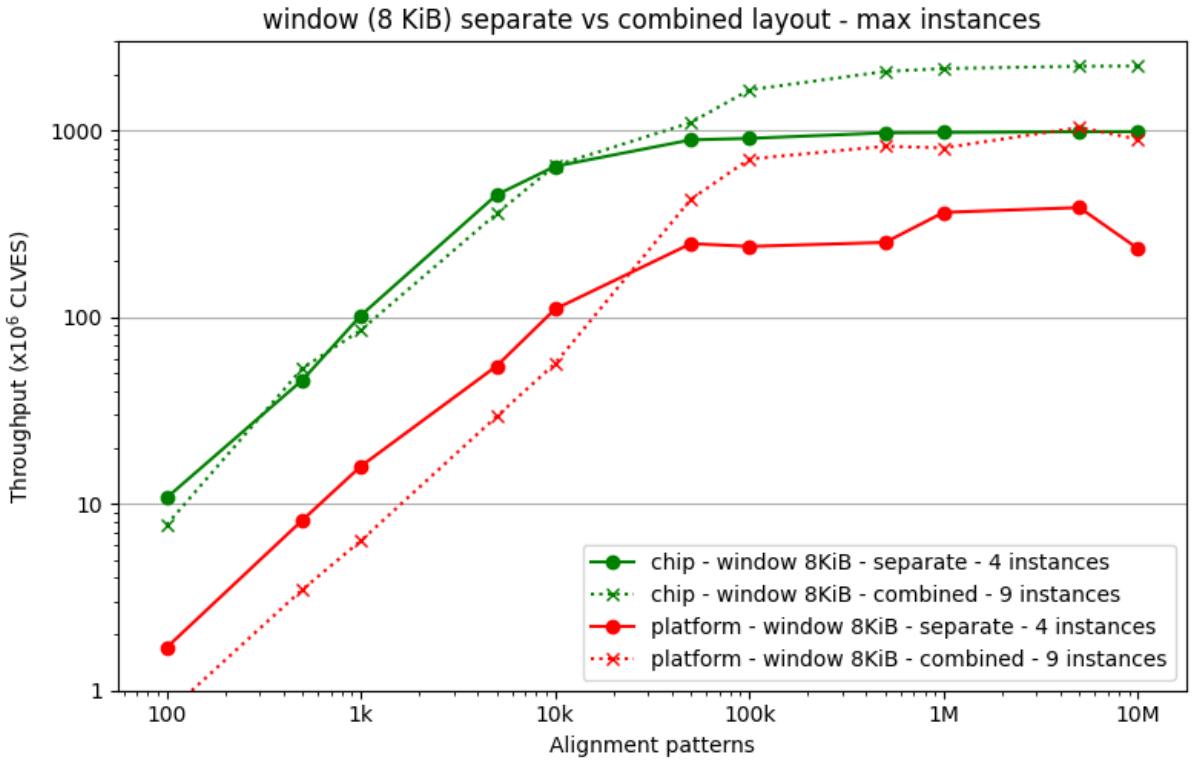
Configuration (interface, layout)	Accel. inst.	1K		10K		100K		1M		10M	
		PL	MEM	PL	MEM	PL	MEM	PL	MEM	PL	MEM
stream, separate	1	104.8	20.9	141.9	89.6	147.3	137.0	147.9	145.6	147.9	147.5
	2	130.2	16.7	259.8	110.6	290.5	194.5	295.4	174.2	295.8	268.7
	4	137.1	20.2	363.6	112.5	563.9	280.0	588.5	338.1	591.3	315.0
stream, combined	1	102.4	20.2	142.4	76.0	147.3	108.0	147.8	109.8	147.9	114.1
	2	133.2	15.6	267.5	95.7	291.2	189.1	295.5	211.8	295.8	224.8
	4	115.5	13.9	415.5	95.0	560.9	299.3	589.1	270.7	591.4	408.6
	8	90.9	9.4	512.8	57.8	1008.5	559.5	1166.9	490.9	1181.4	566.6
	9	88.3	10.4	615.7	68.3	1184.6	504.5	1309.3	520.2	1328.3	583.1
1-KiB window, separate	1	149.0	20.6	215.3	95.7	227.5	159.0	228.4	137.1	228.5	170.0
	2	163.6	19.2	361.0	128.7	448.2	256.1	456.3	221.4	457.0	306.7
	4	137.2	17.3	579.3	120.5	837.9	255.3	908.2	332.6	913.6	243.3
1-KiB window, combined	1	150.6	21.1	216.0	84.4	227.2	124.3	228.4	130.2	228.6	130.9
	2	155.8	17.4	398.1	101.9	449.8	220.4	456.3	228.0	457.0	260.3
	4	156.3	14.1	581.2	96.8	833.0	316.0	907.7	345.2	913.6	248.0
	8	117.6	13.2	643.6	91.1	1408.5	575.9	1780.8	598.3	1823.5	582.5
	9	81.2	6.6	602.1	68.2	1538.3	684.7	1996.4	583.5	2050.3	1003.4
8-KiB window, separate	1	153.9	22.9	234.4	78.3	245.5	108.9	246.9	115.8	247.1	115.0
	2	93.9	19.3	316.4	95.6	483.6	143.6	493.1	166.2	494.0	201.0
	4	102.1	15.9	642.8	110.4	907.5	239.2	976.7	364.1	987.1	233.0
8 KiB window, combined	1	150.1	22.1	228.9	86.3	245.5	131.4	246.9	137.3	247.1	140.4
	2	148.3	19.7	426.0	108.9	483.2	232.8	492.9	266.9	494.1	271.2
	4	164.8	13.1	604.1	96.9	909.0	386.0	980.7	396.1	987.6	467.5
	8	86.0	8.8	593.5	67.8	1565.4	658.6	1925.0	672.9	1971.5	913.9
	9	85.2	6.3	648.4	56.4	1651.5	703.3	2149.0	806.9	2216.4	898.1
16-KiB window, separate	1	145.7	18.1	232.4	73.1	246.9	108.4	248.3	114.4	248.5	115.1
	2	187.4	14.7	422.6	93.2	488.1	175.4	496.0	125.3	496.9	134.9
	4	131.3	10.6	533.0	89.3	909.8	280.2	988.3	227.6	993.2	226.1
16-KiB window, combined	1	138.1	18.4	225.9	79.3	246.9	118.8	248.4	125.3	248.5	121.2
	2	94.7	14.2	423.4	96.6	483.6	200.0	495.7	231.7	496.9	246.3
	4	79.3	10.3	627.0	82.2	928.3	313.6	985.2	383.0	992.7	374.9
	8	59.4	6.1	643.9	54.1	1382.3	380.9	1922.8	509.1	1980.6	494.2
	9	63.6	5.4	651.0	51.6	1599.2	456.0	2141.1	539.3	2228.9	532.4

Results - multiple instances

Combined and separate scale similarly

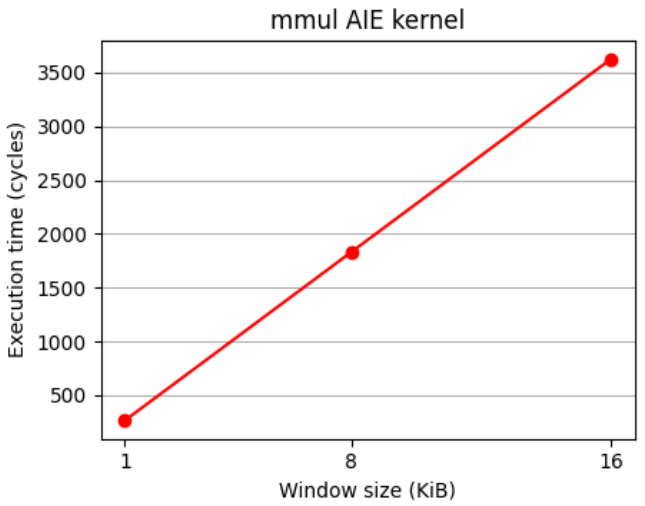
Combined can scaler further with 9 instances

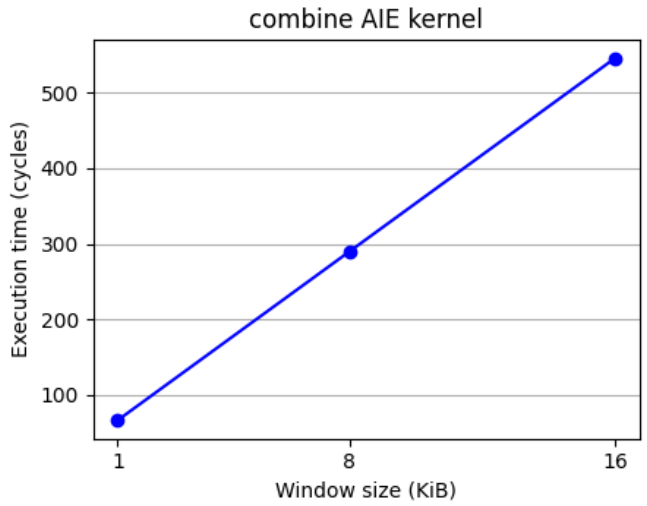
Short CLVs see slowdown for more instances



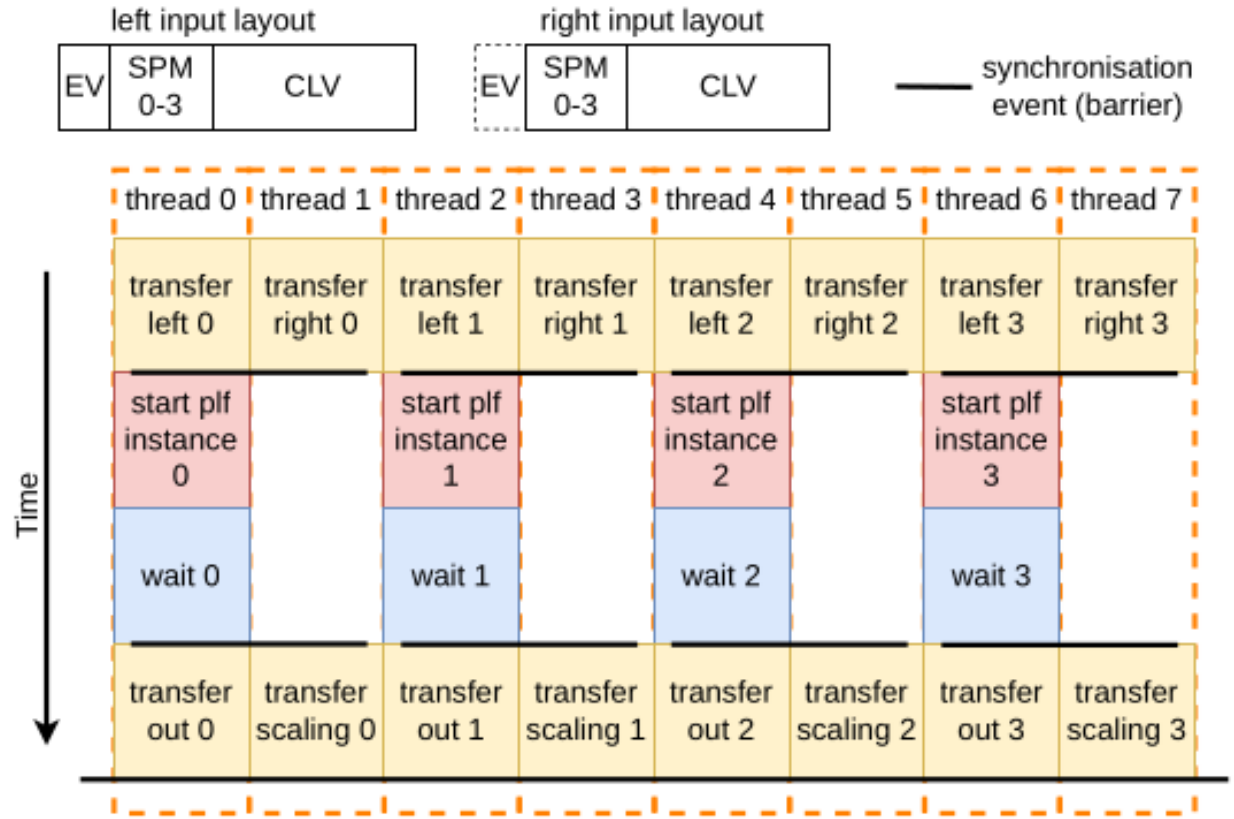
Window size effect on execution time

Size (KiB)	mmul Sep.	mmul Com.	combine Sep.	combine Com.	ev Sep.	ev Com.
1	266	258	66	67	266	254
8	1834	1826	290	291	1834	1822
16	3626	3597	546	544	3626	3626

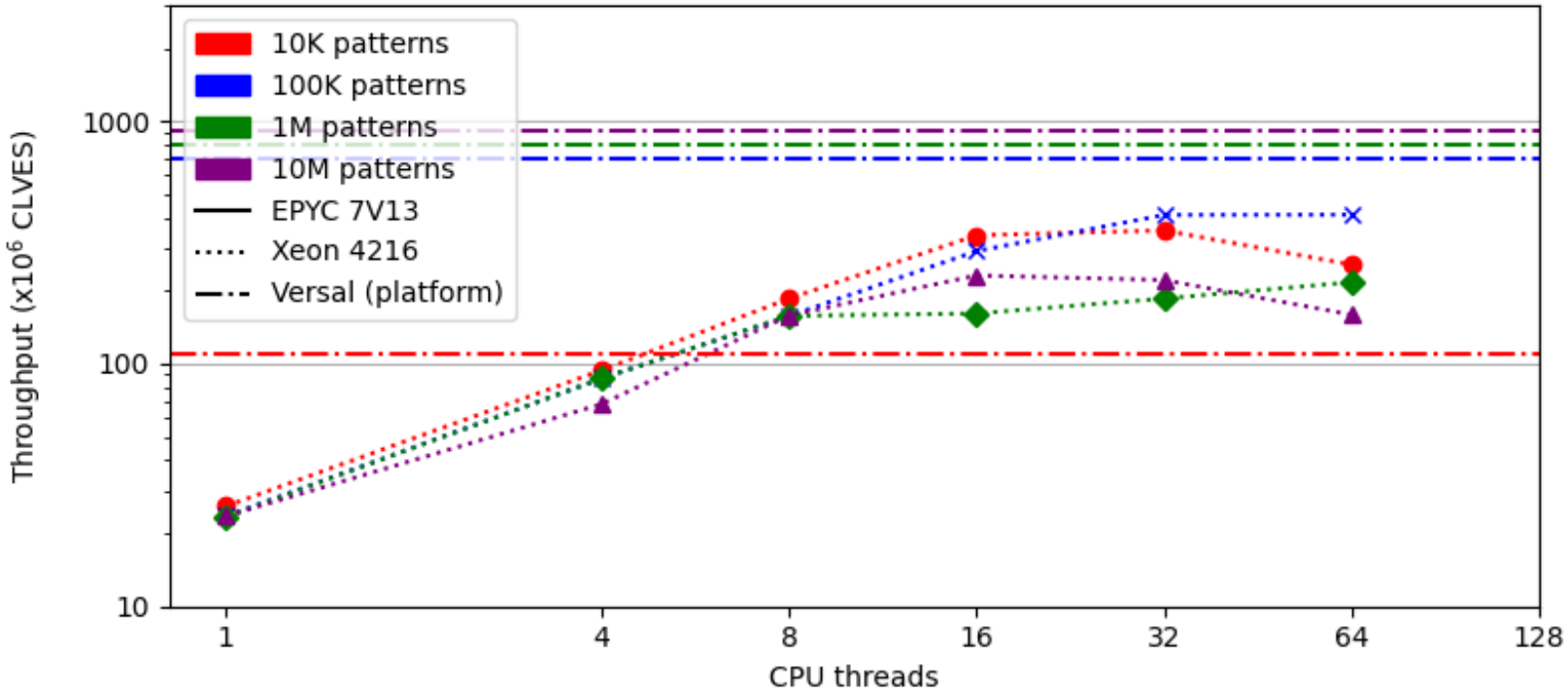




Host Program



RAxML PLF on CPU

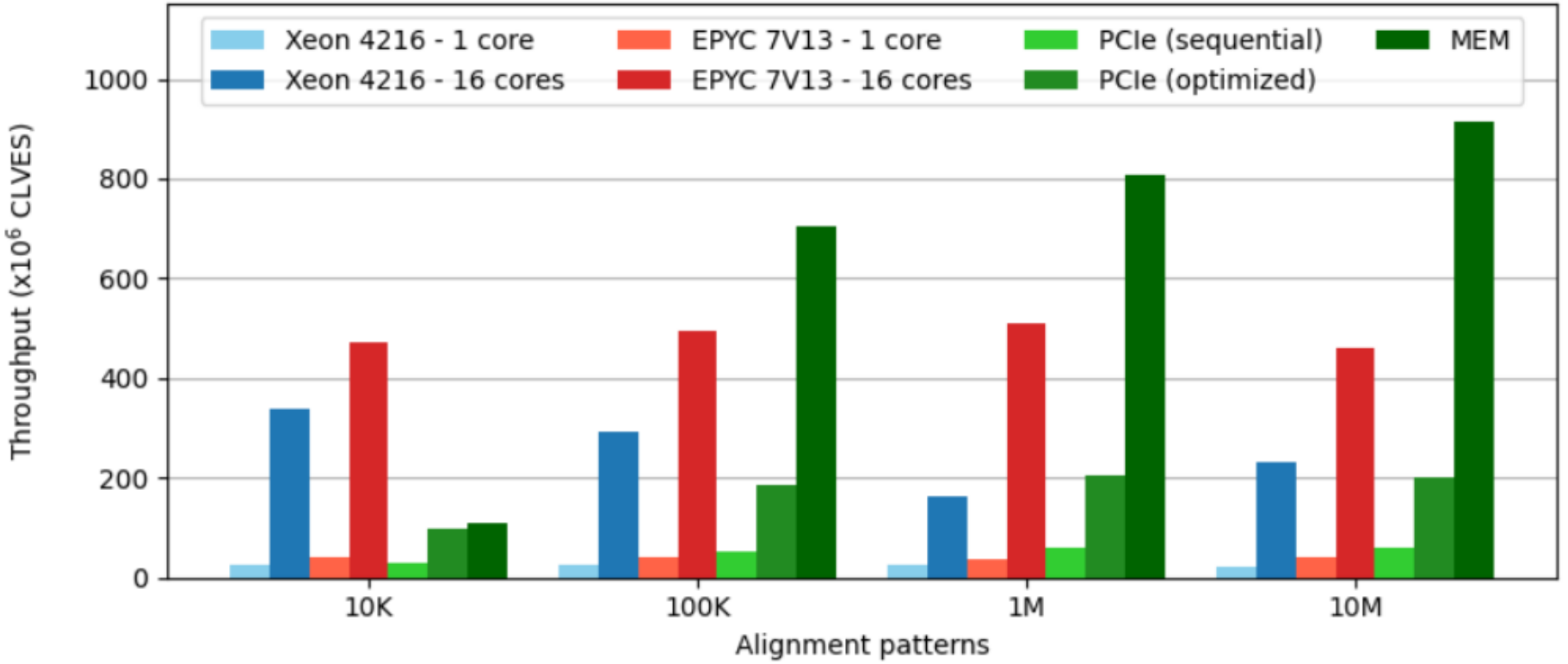


Performance comparison - CPU

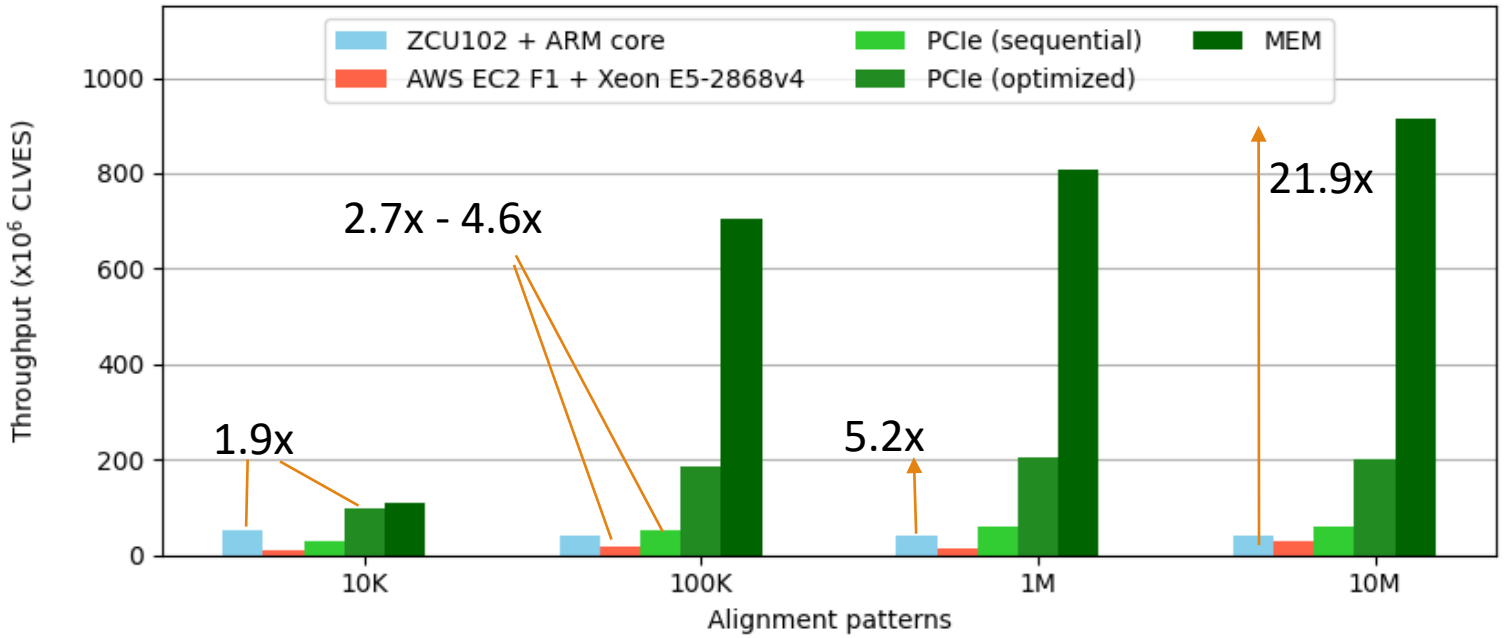
System (optimized)
-Up to 5.6x (1 EPYC)
-Up to 10.6x (1 Xeon)

Platform
-Up to 23.8x (1 EPYC)
-Up to 47x (1 Xeon)

-Up to 2x (16 EPYC)
-Up to 4x (16x Xeon)



FPGA Performance comparison



Malakonakis, P., Brokalakis, A., Alachiotis, N., Sotiriades, E., & Dollas, A. (2020, October). Exploring modern FPGA platforms for faster phylogeny reconstruction with RAxML. In *2020 IEEE 20th International Conference on Bioinformatics and Bioengineering (BIBE)* (pp. 97-104). IEEE.