Experimental Study of Power Consumption of Basic Parallel Programs

Authors: Roblex NANA TCHAKOUTE and Claude TADONKI

Mines Paris - PSL (CRI - France)

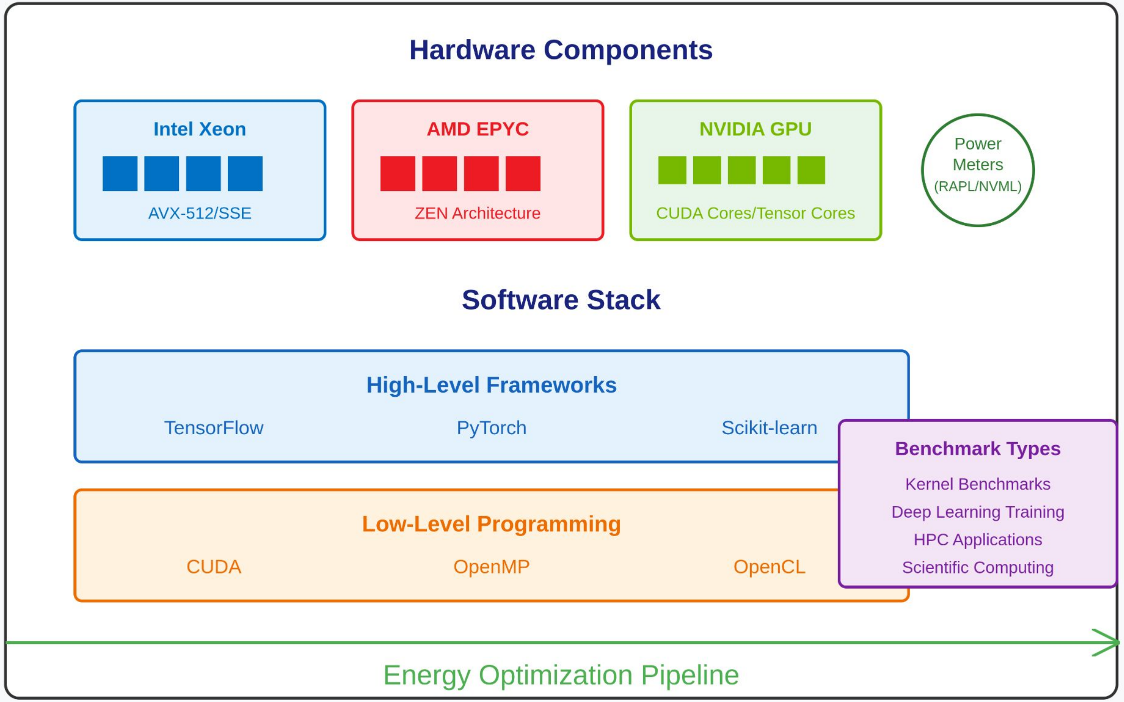
15th Workshop on Applications for Multi-Core Architectures, in conjunction with the
36th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD 2024)
Hawaii, USA, November 13-15, 2024

Context

- Nowadays, computers architecture are increasingly complex *(differents vendors, different microarchitectures, accelerated hardwares, etc….)*
- Applications design must consider this complexity *(don't forget programing paradigms and software stacks. Aka OpenMP, CUDA, Python, Pytorch, etc…)*
- The theoretical complexity of a good algorithm design may be constrained by the target computer architecture for certain applications *(influence of target architecture)*
- Specific architectures are necessary for optimal performance with some applications *(Don't worry, every year, we introduce innovative new versions of devices.)*
- But what is the problem ? Energy/power consumption and related carbon footprint

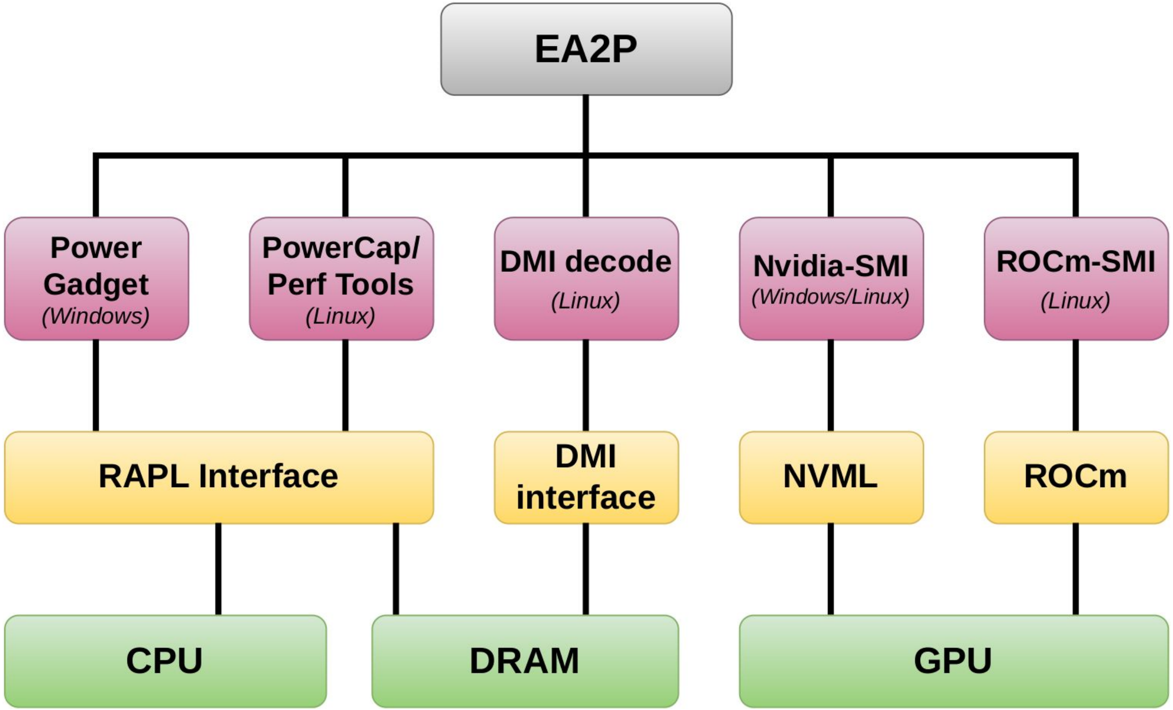
Motivations for benchmarking

- Not every code can take advantage of full computer's resources.
- We need to assess workloads by considering the energy needs of the main devices.
- Fine grained measurements help to understand the behavior per devices and then optimize accordingly



Previous work: Energy Aware Application Profiler (EA2P)

Open Source available at :
https://github.com/HPC-CRI/EA2P
Documentation at :
https://hpc-cri.github.io/EA2P/



Energy can be measured through integrated Power Meters (or sensors)

Research questions

- To what extent does execution time correlate with energy consumption ?
- How data traffic on memory can affect energy/power ?
- Does SIMD use more energy than scalar implementation ?
    - energy/power & time acceleration
- What about parallel multithreaded versions ?
    - Cores/Threads scalability (acceleration) for time & energy correlation
    - usage of full power of the CPU (SIMD+OpenMP all core)
- And what behavior for GPU (with another level of complexity for python libs) ?

Benchmarking Methodology

1. Kernel Selection
→ Compute Bound Kernels
→ Memory Bound Kernels

2. Implementation Versions
- Sequential (Basic)
- Vectorized (SSE/AVX)
- Multithreaded (OpenMP)
- GPU (DL Frameworks)

3. Code Instrumentation
- LIKWID MARKER API (CPU)
- EA2P (GPU)

4. Dataset Configuration
→ Framework-specific sizes
→ Platform-dependent adjustments

5. Compilation Setup
→ Control optimization flags
→ Prevent unexpected optimizations

6. Configure Measurements
→ LIKWID events
→ Automation scripts
→ Default configurations

7. Execute Experiments
→ Iterate through event groups & code versions

8. Analysis
→ Preprocess results and discuss findings

Workloads selection

- **TRIAD (saxpy):** A memory bound kernel and common operation in linear algebra with various applications in computer science, and many other fields.

$$C_i = \alpha \times A_i + B_i$$

- **gemm:** Compute bound. This formula represents the calculation of each element of the resulting matrix C based on the elements of matrices A and B.

$$C_{ij} = \sum_{k=0}^{N-1} A_{ik} \cdot B_{kj}$$

- **Distance:** Generalized form of the Euclidean distance, where instead of computing the distance between two points (x, y), you are summing the distances between the origin (0, 0) and each point (xi, yi)

$$\sum_{i=1}^{n} \sqrt{(x_i)^2 + (y_i)^2}$$

- **spmv:** Memory and latency bound kernel. COO data format is used for simplicity

$$y_{\text{rowind}[i]} += \text{val}[i] \cdot x_{\text{colind}[i]}$$

- **7-point 3D stencil**: a mixed-bound kernel, commonly used in the modeling of physical phenomena like heat diffusion and fluid dynamics.

$$L(i,j,k) = -6 \cdot u(i,j,k) + u(i+1,j,k) + u(i-1,j,k) + u(i,j+1,k) + u(i,j-1,k) + u(i,j,k+1) + u(i,j,k-1)$$

- **Monte Carlo Pi Estimation**: also a compute-bound kernel that is widely used in stochastic simulations and financial modeling.

$$\begin{cases} x_i^2 + y_i^2 \le 1 \\ \pi \approx 4 \times \frac{M}{N} \end{cases}$$

M is the number of points inside the circle and N is the total number of points generated.

Platform Characteristics

Name	Chirop	Chuc
CPU model	Intel Platinum 8358	AMD EPYC 7513
Clock speed	2.6 GHz	2.6 GHz
Turbo Speed	Up to 3.4 GHz	Up to 3.65 GHz
Physical Cores	32 (Threads: 64)	32 (Threads: 64)
L1 iCache	1,024KB 8-way set	1,024KB 8-way set
L1 dCache	1,536KB 12-way set	1,024KB 8-way set
L2 Cache	40MB 20-way set	16MB 8-way set
L3 Cache	48MB 12-way set	128MB 16-way set
DRAM Memory	512GB DDR4-3200	512GB DDR4-3200
CPU TDP	250W	200W
GPU Model	/	Nvidia A100 SXM4
GPU TDP	/	400W
GPU Memory	/	40GB HBM2
Data precision	Single	Single
SIMD extensions	SSE, AVX, AVX512	SSE, AVX
Operating System	Linux Debian 5.10.209	Linux Debian 5.10.209

Idle power consumption analysis *(when no program is running)*

runtime [s]	1	2	4	8	16	32	64
AMD							
power PKG [W]	57.73	55.60	55.47	55.62	56.52	56.78	56.02
energy PKG [J]	57.74	111.22	221.89	444.98	904.45	1817.08	3585.92
Intel							
power PKG [W]	52.22	49.11	49.68	49.07	48.48	47.64	48.10
energy PKG [J]	52.22	98.22	198.71	392.56	775.74	1524.49	3078.67
power RAM [W]	9.97	9.69	9.64	9.60	9.41	9.42	9.43
energy RAM [J]	9.97	19.38	38.55	76.78	150.53	301.36	603.49

Idle energy/power for single core usage through system runtime

$$\begin{cases} E_{idle}(t) = \alpha + \beta \times t \\ E_{application} = E_{measured} - E_{idle}(t_{application}) \end{cases}$$

Idle energy modeling and real application energy deduction

$$\begin{cases} E_{idle-CPU-Intel}(t) = 3.87 + 48.36 \times t \\ E_{idle-RAM-Intel}(t) = -0.0003 + 9.67 \times t \\ E_{idle-CPU-AMD}(t) = 1.67 + 56.50 \times t \end{cases}$$

Idle energy for single core usage. *With :*

- *RMSE = 6.36 and R^2 = 1.00 for Intel CPU;*
- *RMSE = 0.49 and R^2 = 1.00 for Intel RAM;*
- *RMSE = 8.65 and R^2 = 0.9999 for AMD CPU*

$$\begin{cases} E_{idle-CPU-Intel}(t) = 4.44 + 47.95 \times t \\ E_{idle-RAM-Intel}(t) = 0.63 + 9.41 \times t \\ E_{idle-CPU-AMD}(t) = 1.88 + 56.14 \times t \end{cases}$$

Idle energy for multicore core usage. *With :*

- *RMSE = 6.23 and R^2 = 0.9977 (Intel CPU);*
- *RMSE = 0.56 and R^2 = 0.9995 (Intel RAM);*
- *RMSE = 3.75 and R^2 = 0.9994 (AMD CPU)*

Intel Ice Lake SP CPU consumes approximately 50W on average in idle state, which represents 20% of its theoretical TDP (250W). The AMD Zen 3 CPU consumes around 56W, which is about 28% of its TDP (200W).

Time and energy measurements of SIMD (vectorisation)

code ver.	time [s]	acc.	energy CPU [J] PKG	app.	CPU acc. PKG	acc. app.	energy RAM [J] tot.	app.	RAM acc. tot.	acc. app.
					SPMV_COO					
seq	9.91	1.00	991.37	476.01	1.00	1.00	164.17	70.02	1.00	1.00
sse	10.74	0.92	1070.30	511.83	0.93	0.93	175.74	73.72	0.93	0.95
avx	9.17	1.08	920.92	443.84	1.08	1.08	153.05	65.89	1.07	1.06
					DIST					
seq	6.45	1.00	663.17	340.64	1.00	1.00	94.83	33.55	1.00	1.00
sse	4.52	1.43	453.75	227.94	1.46	1.49	66.98	24.08	1.42	1.39
avx	2.49	2.59	253.93	129.32	2.61	2.63	38.16	14.49	2.48	2.32
					GEMM					
seq	116.56	1.00	11943.84	6116.03	1.00	1.00	1606.94	499.66	1.00	1.00
sse	56.08	2.08	5662.58	2858.58	2.11	2.14	774.71	241.95	2.07	2.07
avx	29.14	4.00	2920.22	1463.07	4.09	4.18	401.29	124.44	4.00	4.02
					TRIAD					
seq	5.43	1.00	544.94	273.55	1.00	1.00	85.12	33.56	1.00	1.00
sse	3.38	1.61	342.35	173.20	1.59	1.58	55.58	23.44	1.53	1.43
avx	2.30	2.36	237.01	122.15	2.30	2.24	39.79	17.97	2.14	1.87

a) Intel

Code version	Time [s]	acc. (time)	Energy PKG[J]	Energy E_{app}	acc. PKG	acc. E_{app}
			SPMV_COO			
seq	14.26	1.00	1108.50	328.50	1.00	1.00
sse	15.22	0.94	1157.10	307.77	0.96	1.07
avx	11.32	1.26	897.72	259.14	1.23	1.27
			TRIAD			
seq	7.17	1.00	547.18	144.32	1.00	1.00
sse	5.84	1.23	433.66	106.94	1.26	1.35
avx	3.19	2.25	251.31	71.77	2.18	2.01
			GEMM			
seq	124.83	1.00	9160.88	2201.11	1.00	1.00
sse	46.95	2.66	3469.80	841.02	2.64	2.62
avx	29.02	4.30	2151.76	518.80	4.26	4.24
			DIST			
seq	3.44	1.00	289.16	92.74	1.00	1.00
sse	2.96	1.16	268.70	100.46	1.08	0.92
avx	1.70	2.03	161.99	64.63	1.79	1.43

b) AMD

Both Intel and AMD show similar behavior across versions, except the euclidean dist computation on AMD with noticeable gap between energy acceleration and time acceleration

Time and energy measurements of Multicore (multithreading)

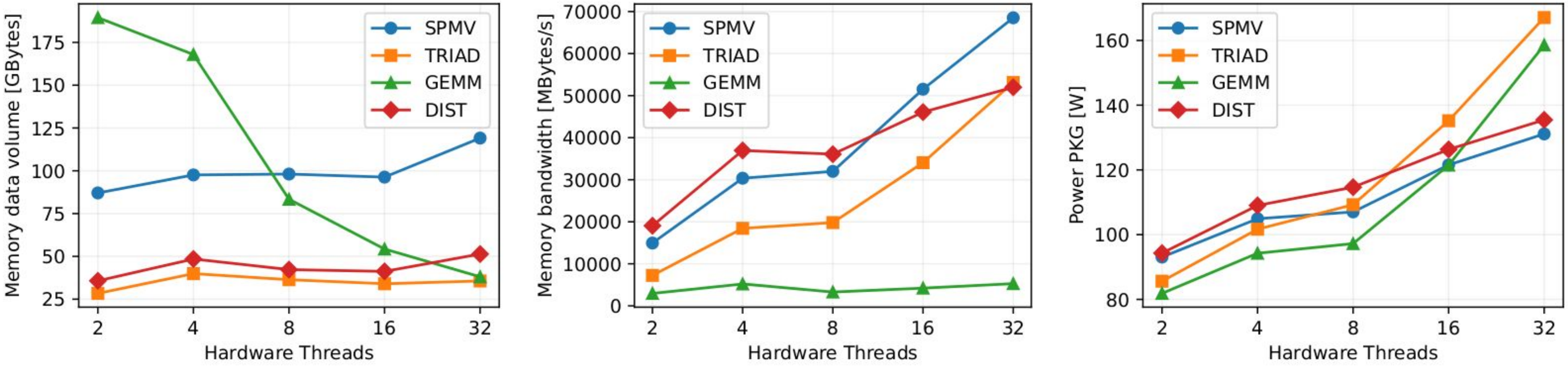
#th	time [s]	time acc.	energy CPU [J] PKG	energy CPU [J] app.	CPU acc. PKG	acc. app.	energy RAM [J] tot.	energy RAM [J] app.	RAM acc. tot	acc. app.
SPMV_COO										
seq	9.91	1.00	991.37	476.01	1.00	1.00	164.17	70.02	1.00	1.00
2	5.63	1.76	597.85	305.27	1.66	1.57	105.59	52.14	1.55	1.34
4	2.96	3.35	349.97	196.12	2.83	2.44	66.35	38.24	2.47	1.83
8	1.66	5.97	230.20	143.83	4.31	3.32	46.37	30.59	3.54	2.29
16	0.99	9.98	175.81	124.16	5.64	3.85	34.74	24.81	4.73	2.82
32	0.71	14.03	159.36	115.57	6.22	4.14	29.40	22.33	5.58	3.14
DIST										
seq	6.45	1.00	663.17	340.64	1.00	1.00	94.83	33.55	1.00	1.00
2	3.34	1.93	370.88	203.91	1.79	1.67	51.59	19.86	1.84	1.69
4	1.69	3.82	219.11	134.72	3.03	2.53	28.37	12.33	3.34	2.72
8	0.88	7.34	144.69	100.76	4.58	3.38	16.51	7.72	5.75	4.35
16	0.50	12.86	111.27	85.20	5.96	4.00	10.99	5.98	8.63	5.61
32	0.41	15.72	95.19	69.80	6.97	4.88	10.14	6.03	9.36	5.56
GEMM										
seq	116.56	1.00	11943.84	6116.03	1.00	1.00	1606.94	499.66	1.00	1.00
2	56.84	2.05	6162.93	3320.87	1.94	1.84	786.53	246.54	2.04	2.03
4	34.26	3.40	4092.08	2378.91	2.92	2.57	473.73	148.22	3.39	3.37
8	17.14	6.80	2471.08	1614.09	4.83	3.79	236.93	74.10	6.78	6.74
16	8.61	13.54	1670.17	1239.89	7.15	4.93	119.88	38.12	13.41	13.11
32	5.56	20.97	1330.47	1052.57	8.98	5.81	77.29	24.49	20.79	20.40
TRIAD										
seq	5.43	1.00	544.94	273.55	1.00	1.00	85.12	33.56	1.00	1.00
2	2.94	1.85	313.14	166.28	1.74	1.65	49.97	22.06	1.70	1.52
4	1.50	3.63	185.50	110.67	2.94	2.47	29.80	15.58	2.86	2.15
8	0.80	6.80	120.49	80.54	4.52	3.40	19.48	11.49	4.37	2.92
16	0.49	11.03	95.21	69.64	5.72	3.93	15.05	10.13	5.65	3.31
32	0.48	11.23	96.20	66.22	5.66	4.13	14.63	9.79	5.82	3.43

a) Intel

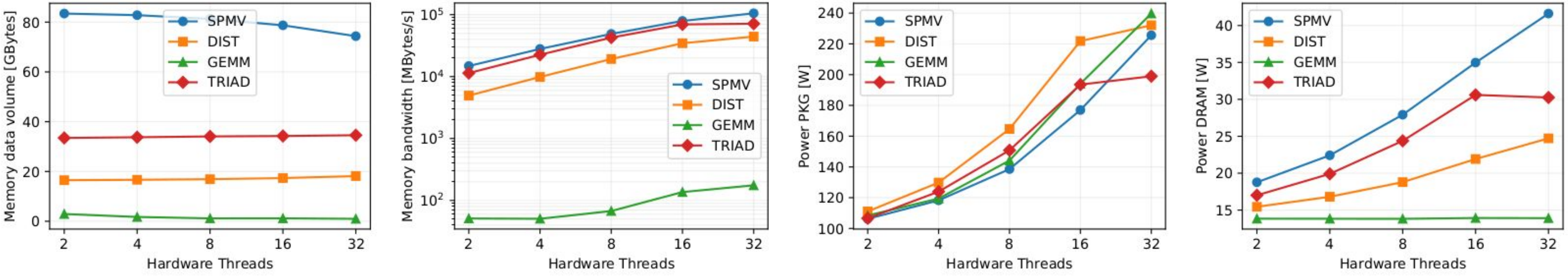
#threads	Time [s]	acc. (time)	Energy PKG[J]	Energy E_{app}	acc. PKG	acc. E_{app}
SPMV_COO						
seq	14.26	1.00	1108.50	328.50	1.00	1.00
2	5.78	2.47	544.11	214.29	2.04	1.53
4	3.22	4.43	338.55	157.49	3.27	2.09
8	3.07	4.64	327.58	156.87	3.38	2.09
16	1.87	7.65	227.27	122.29	4.88	2.69
32	1.63	8.72	229.85	122.81	4.82	2.67
TRIAD						
seq	7.17	1.00	547.18	144.32	1.00	1.00
2	3.87	1.85	334.74	114.63	1.63	1.26
4	2.13	3.36	215.68	97.38	2.54	1.48
8	1.82	3.93	200.93	97.04	2.72	1.49
16	0.99	7.26	135.64	74.24	4.03	1.94
32	0.68	10.60	110.45	71.09	4.95	2.03
GEMM						
seq	124.83	1.00	9160.88	2201.11	1.00	1.00
2	64.15	1.95	5248.30	1658.71	1.75	1.33
4	32.22	3.87	3039.53	1232.53	3.01	1.79
8	25.44	4.91	2468.35	1049.06	3.71	2.10
16	12.98	9.62	1562.17	849.76	5.86	2.59
32	7.14	17.49	1136.23	732.24	8.06	3.01
DIST						
seq	3.44	1.00	289.16	92.74	1.00	1.00
2	2.03	1.70	177.53	75.66	1.63	1.23
4	1.33	2.59	142.91	69.05	2.02	1.34
8	1.15	2.99	136.36	66.27	2.12	1.40
16	0.90	3.81	118.28	62.61	2.44	1.48
32	0.98	3.50	132.67	77.22	2.18	1.20

b) AMD

Memory performance and power consumption



a) AMD



b) Intel

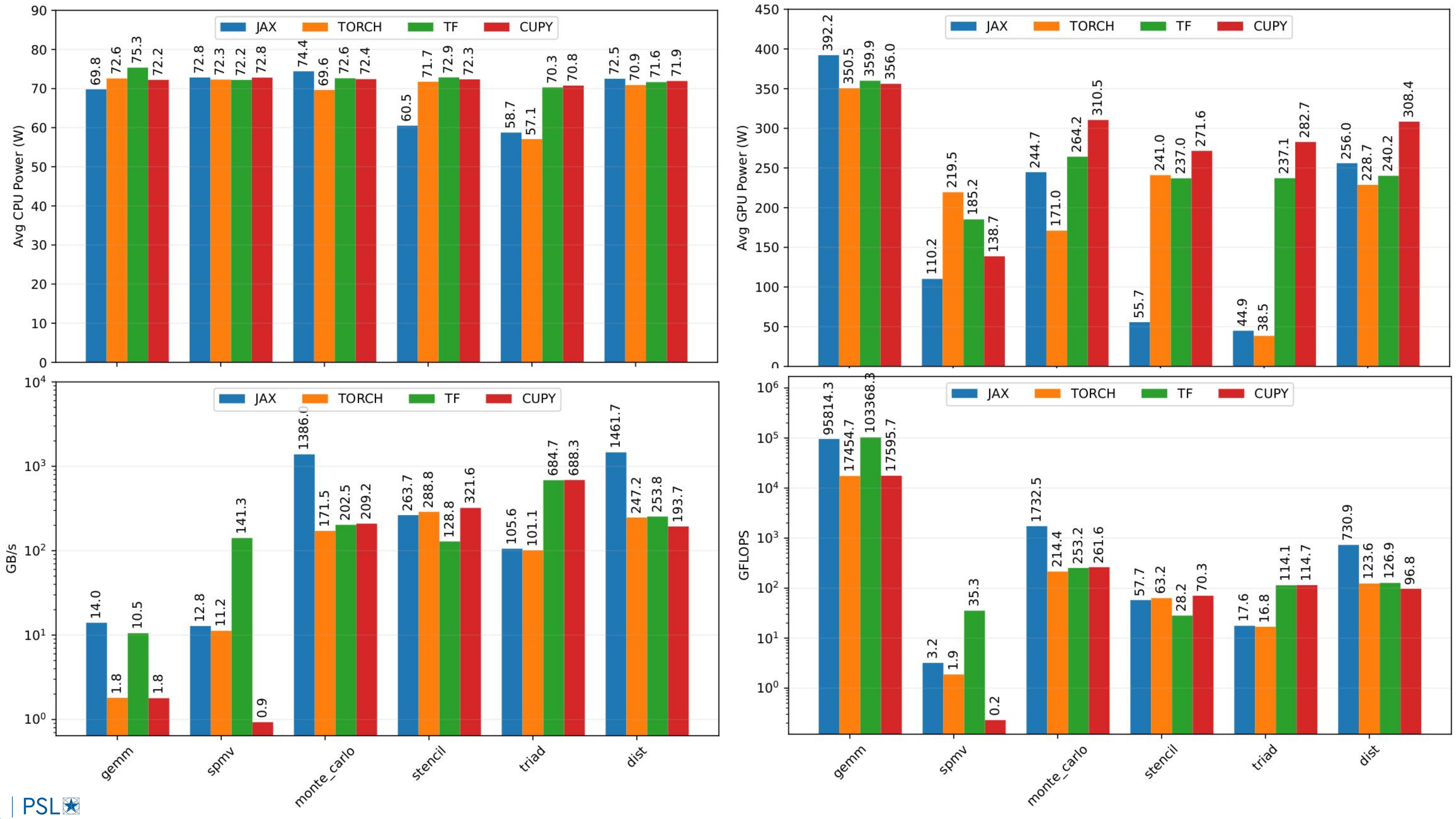
Overall insights (CPU)

- The correlation between time and energy seems to remain valid in all scenarios
- SIMD vectorization generally result in a correlation between execution time acceleration and energy efficiency by the similar factor
- While potentially multithreading improves the computing speed, it does not (always) scale linearly with power consumption,
- Higher bandwidth is likely to yield an increase of the power draw but can improve the overall energy efficiency, particularly with large data volume
- The power consumption of CPUs does not always reflect the memory or compute-bound nature of an application

Benchmark results from the libraries standpoint (Nvidia GPU)

Bench	lib	CPU (J)	GPU (J)	GPU Power (W)	Time (s)	Gflops/s	Gflops/W
Stencil	torch	685.95	2304.11	241.01	9.56	63.18	0.262
	tflow	904.74	2942.69	236.93	12.42	28.17	0.119
	jax	1238.48	1139.35	55.66	20.47	57.68	1.036
	cupy	621.10	2331.35	271.08	8.59	70.34	0.259
DIST	torch	458.01	1477.74	228.75	6.46	123.62	0.540
	tflow	450.81	1511.22	240.25	6.29	126.91	0.528
	jax	1189.62	4199.76	255.93	16.41	730.87	2.856
	cupy	445.36	1909.11	308.42	6.19	96.84	0.314
TRIAD	torch	1354.18	913.21	38.48	23.73	16.85	0.295
	tflow	245.76	828.76	236.79	3.50	114.12	0.482
	jax	1333.95	1019.16	44.87	22.71	17.61	0.392
	cupy	246.03	983.17	282.52	3.48	114.71	0.406
SPMV	torch	1928.21	5853.35	219.47	26.67	1.88	0.008
	tflow	246.71	632.71	185.00	3.42	35.32	0.190
	jax	328.25	496.71	110.13	4.51	3.20	0.029
	cupy	1712.68	3263.85	138.71	23.53	0.23	0.002
Monte carlo	torch	486.14	1194.05	171.06	6.98	214.44	1.253
	tflow	572.97	2084.31	264.17	7.89	253.18	0.958
	jax	428.58	1409.75	244.75	5.76	1732.50	7.079
	cupy	414.57	1777.49	310.21	5.73	261.56	0.843
GEMM	torch	1621.75	7832.08	350.43	22.35	17454.74	49.810
	tflow	298.63	1426.65	360.26	3.96	103368.22	286.923
	jax	200.24	1124.70	391.88	2.87	95814.29	244.498
	cupy	1685.32	8309.03	355.00	23.34	17595.68	49.426

GFLOPS and GB/s vs Power all benchmarks and frameworks (Nvidia GPU)



Overall insights (GPU)

- **JAX:** Demonstrates its strength not only on compute-bound tasks but also on complex memory-bound operations like 3D-stencil.
- **TensorFlow**: While TensorFlow show good performances in a broad range of tasks, it appears less efficient in a complex memory-bound scenario like 3D-stencil.
- **PyTorch**: Memory-bound tasks are still a little challenging. However, it remains a versatile framework, particularly for tasks with large data size support.
- **CuPy**: CuPy is globally powerful and can handle highly optimized workloads at the price of higher power consumption especially on memory-bound cases.

Conclusion and future works

- Conclusion:
  - This work underscores the complex relationship between performance, power consumption, and energy efficiency across different architectures and workloads
  - Optimizing for both energy efficiency and performance requires a comprehension of workload characteristics, hardware-specific features, and the impact of parallelization and vectorization
- Future works
  - Apply system power management techniques to study best energy/performance strategies
  - Design energy aware optimization framework using energy patterns from various kernels studied
  - Design power management techniques by deeply investigating all hardware power-saving features for energy-aware programming and scheduling

Thank you for your Attention !

Email : {roblex.nana_tchakoute, claude.tadonki}@minesparis.psl.eu

This research was supported by The Transition Institute 1.5 driven by École des Mines de Paris - PSL

Experiments presented in this paper were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations.