Python-Blosc2
Compress Better, Compute Bigger!

Francesc Alted / @FrancescAlted@masto.social

The Blosc Development Team / @Blosc2@fosstodon.org

CEO [[ ]] ironArray / francesc@ironArray.io

EuroSciPy 2025 -- Kraków, Poland
August 21st 2024

Agenda

- How Physics Laws Affect Data Science
- Compressing Better
- Computing Bigger
- Caterva2: Sharing Faster

Who is ironArray SLU?

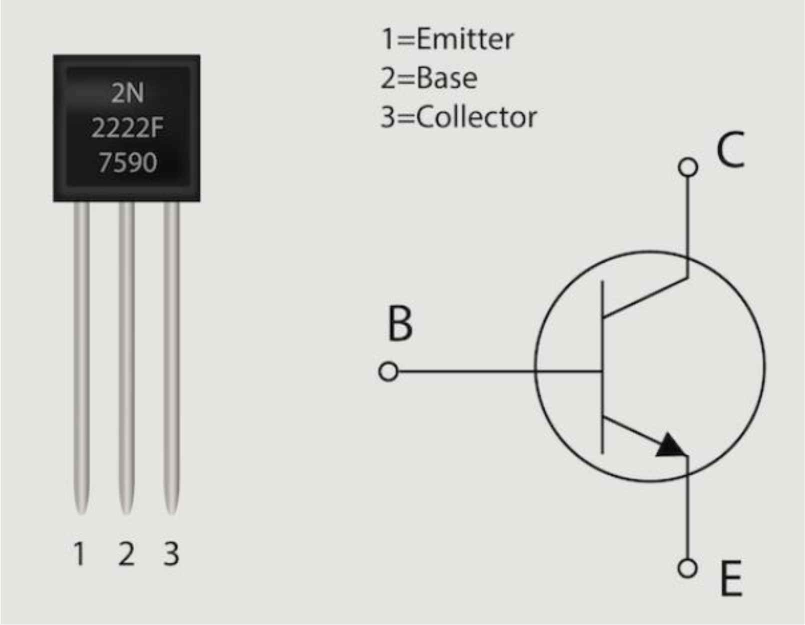
- We are the developers of PyTables, numexpr and Blosc libraries.
- Team of experts empowering you to harness the full potential of compression for big data: we are here to help!

https://ironarray.io

Intro

How Physics Laws Affect Data Science

Energy Consumption of the Transistor



$$\text{Power} \propto V^2 \times f$$

V: voltage

f: frequency

The **Power Wall**: as transistors got smaller, they didn't proportionally reduce their power consumption as expected.

Power Wall Consequences

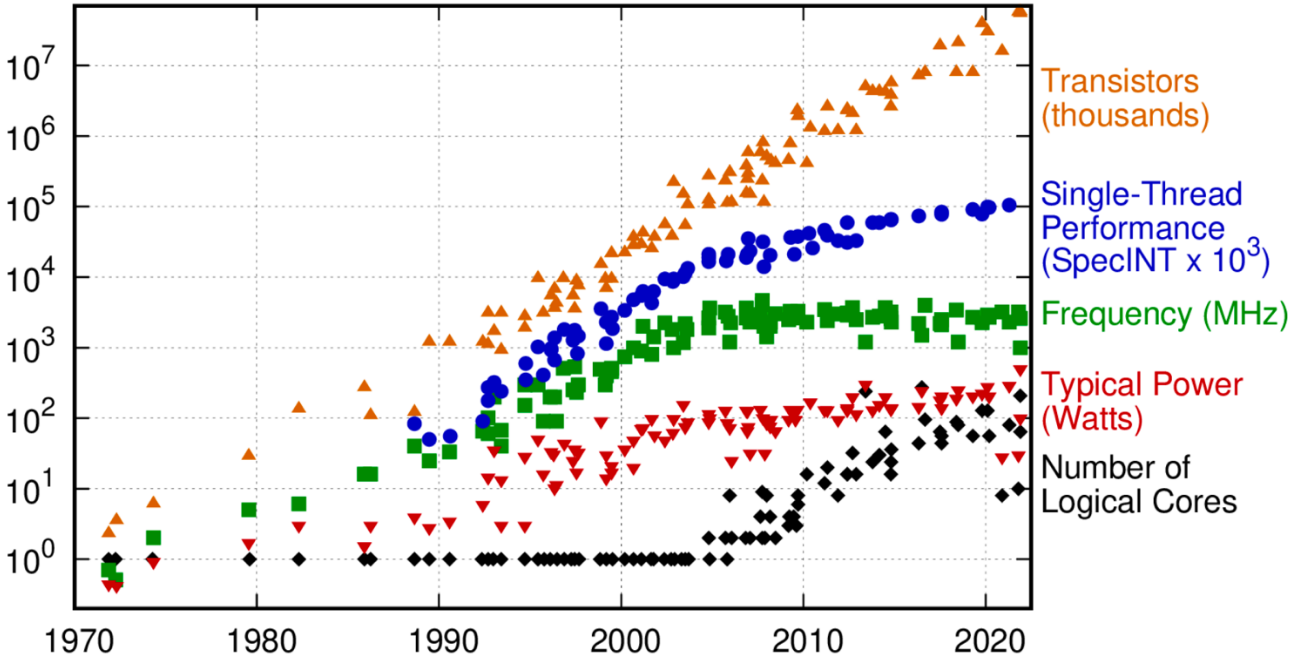


Figure 1.1: 50 Years of Microprocessor Trend Data. © *Image by K. Rupp via karlrupp.net.* Original data up to the year 2010 was collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten. New plot and data collected for 2010-2021 by K. Rupp.

Source:
Performance Analysis and Tuning on Modern CPUs
By Denis Bakhvalov

The Shift to More Transistors

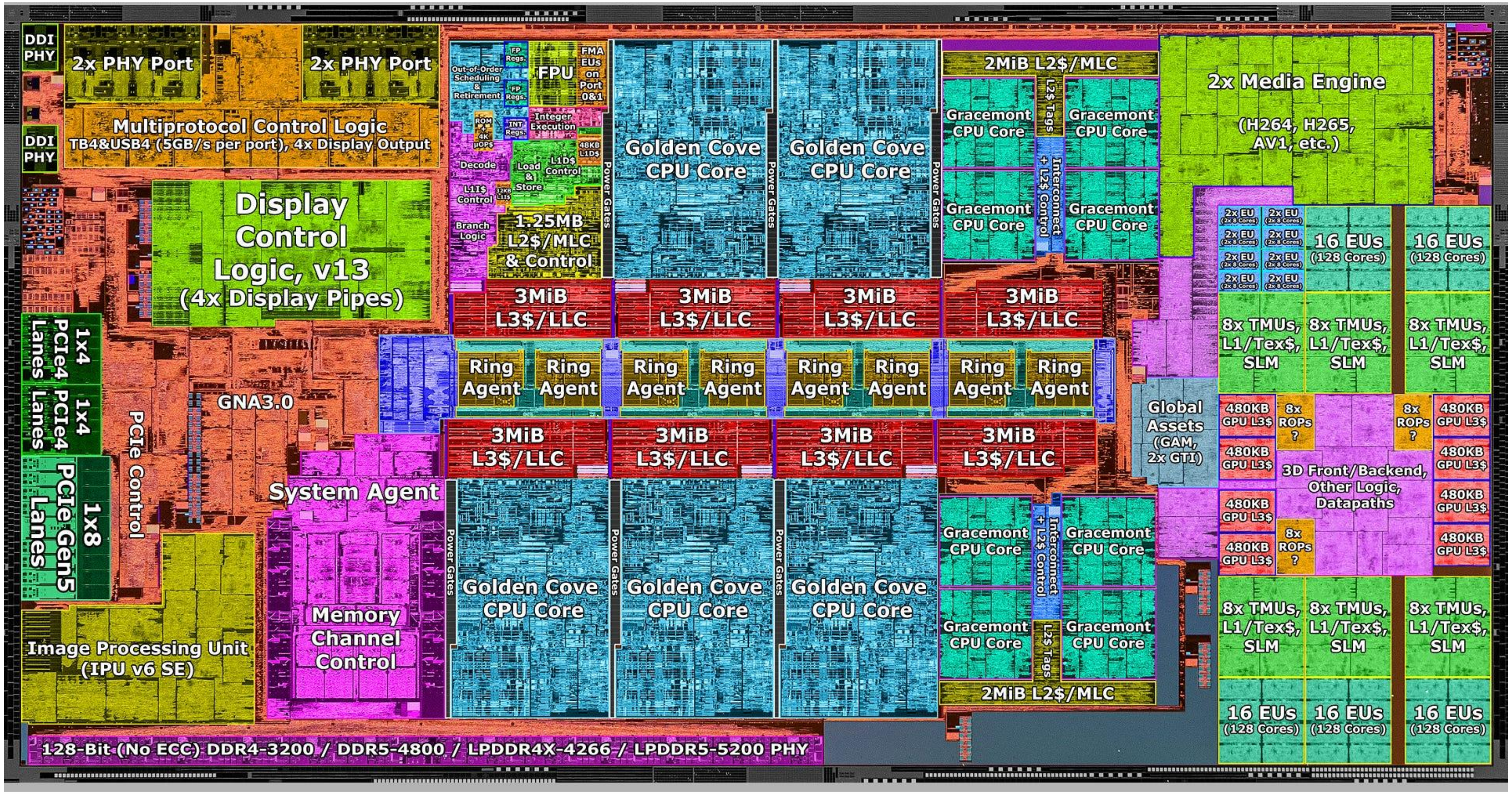
Instead of faster single cores, manufacturers now focus on:

- **Multiple cores** for parallel processing.
- **Specialized processing units** (GPUs, AI accelerators, dedicated media processors).
- **Hardware acceleration** for specific tasks (e.g. compression via Intel QAT).
- **Larger caches** to reduce memory bottlenecks.

CPUs Have Become Complex Beasts!

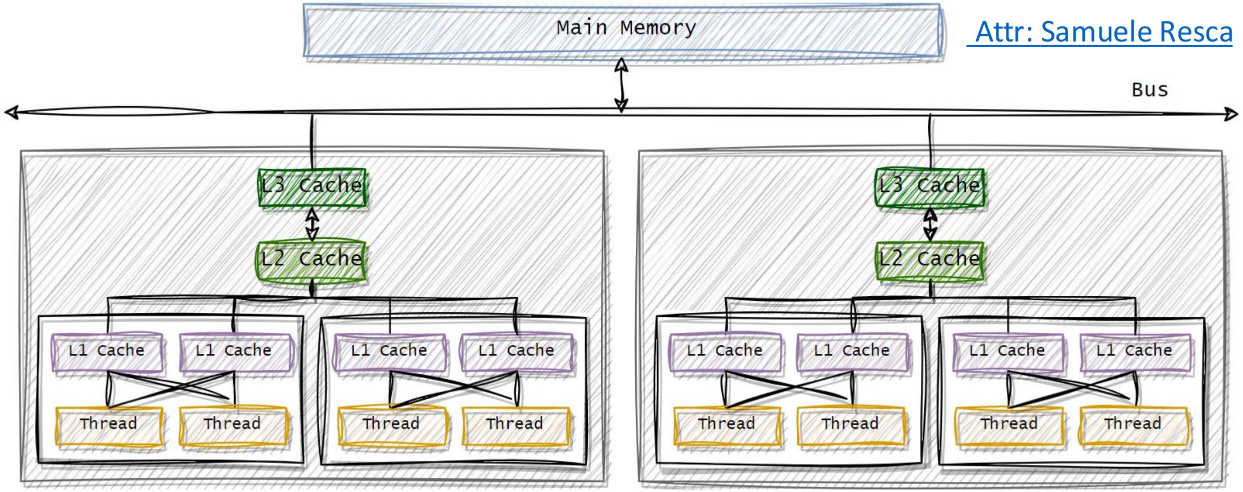
Intel Alder Lake

Source: Alder Lake P (6P + 8E + 96 EU) die shot from Intel on Reddit - https://www.reddit.com/r/intel/comments/se1xxx/today_ama_from_830am_to_300_pm_pst_12th_gen_intel



Slide compiled by Locuza, June 2022

Memory Is Getting More Complex Too!



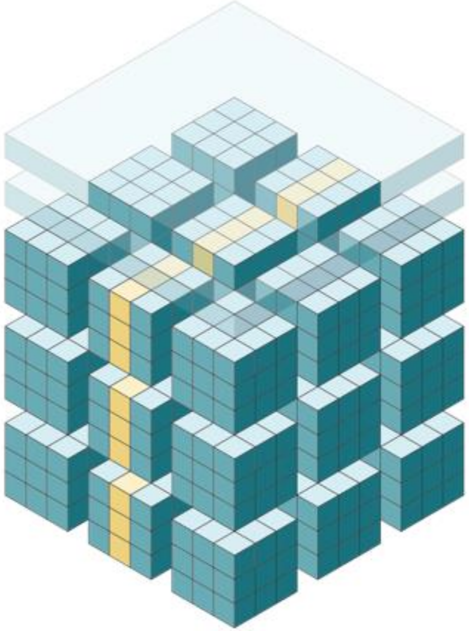
Attr: Samuele Resca

- Memory caches are closer to the CPU and hence, are faster. However, they should be smaller due to power dissipation constraints.
- Efficient access to memory from multi-threaded applications is getting more difficult because interactions among threads and data.

We Live An Ironic Moment In History

- In one hand, we (together with LLMs) are **producing more code** than ever.
- On the other hand, software is increasingly slow; or, if you want, it is **increasingly difficult to get the most out of modern CPUs** and GPUs (because of their complexity).
- Conclusion: We are **producing suboptimal code at a rate never seen before**!

Solution: stand on the shoulders of efficient libraries!



Blosc2

Better compression for multidimensional, binary data

https://www.blosc.org/

What Is Blosc2?

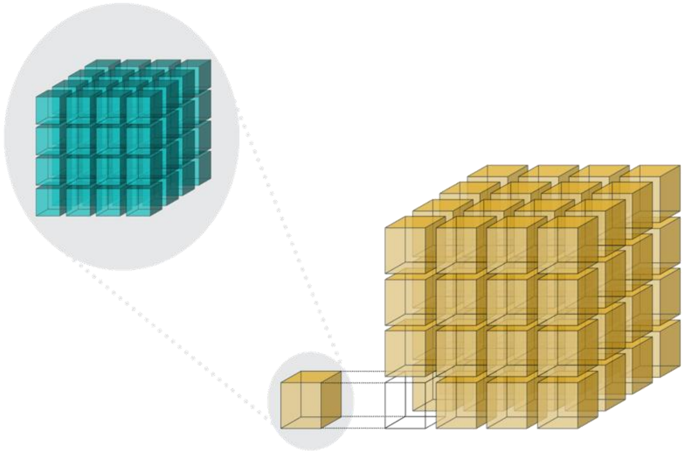
A set of codecs and filters to compress data, orchestrated in a way that **leverages modern computer architecture**:

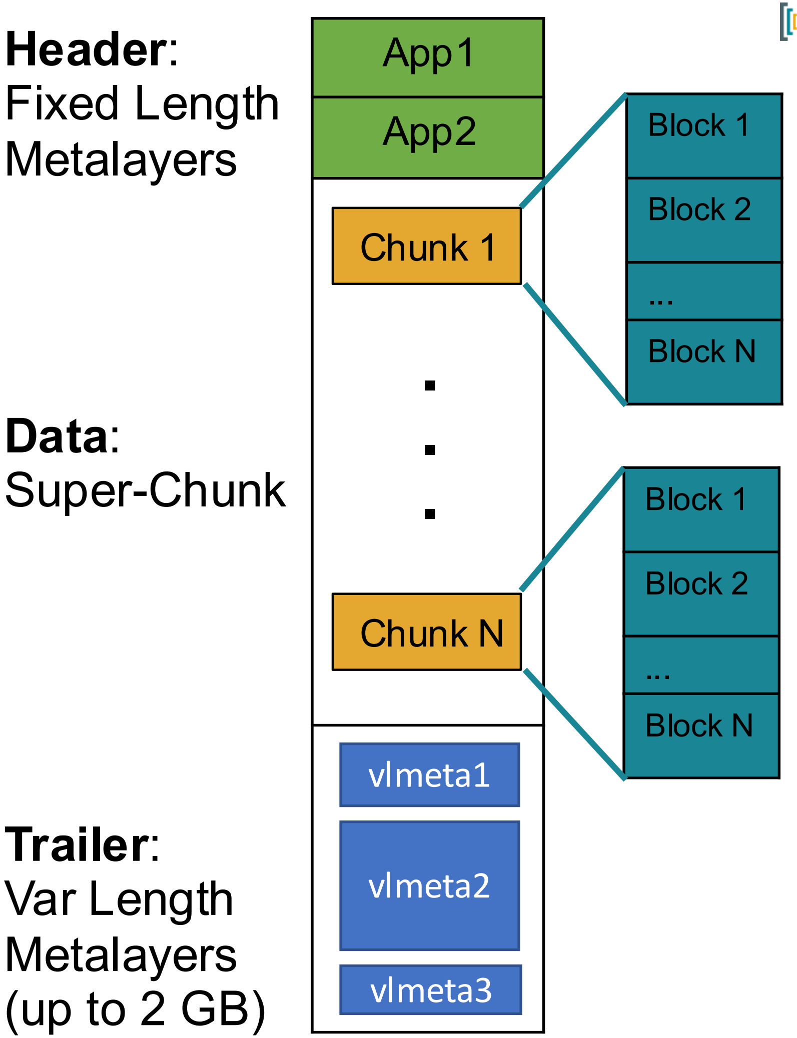
- Support for **multithreading**: use the full cores in your CPU.
- Automatically use of **modern SIMD instructions** (SSE2, AVX2, AVX512, NEON, ALTIVEC) for performance.
- **Double partitioning**, mimicking the multi-level caches in modern CPUs, and adapted to their sizes automatically.
- **In-memory** or **on-disk storage**.

https://github.com/Blosc/c-blosc2

Blosc2 Architecture

- ✓ 64-bit containers
- ✓ Metalayers for adding info for apps and users
- ✓ Blosc2 NDim: Multi-dim blocks and chunks





Header:
Fixed Length
Metalayers

App1

App2

Data:
Super-Chunk

Chunk 1

Block 1

Block 2

...

Block N

Chunk N

Block 1

Block 2

...

Block N

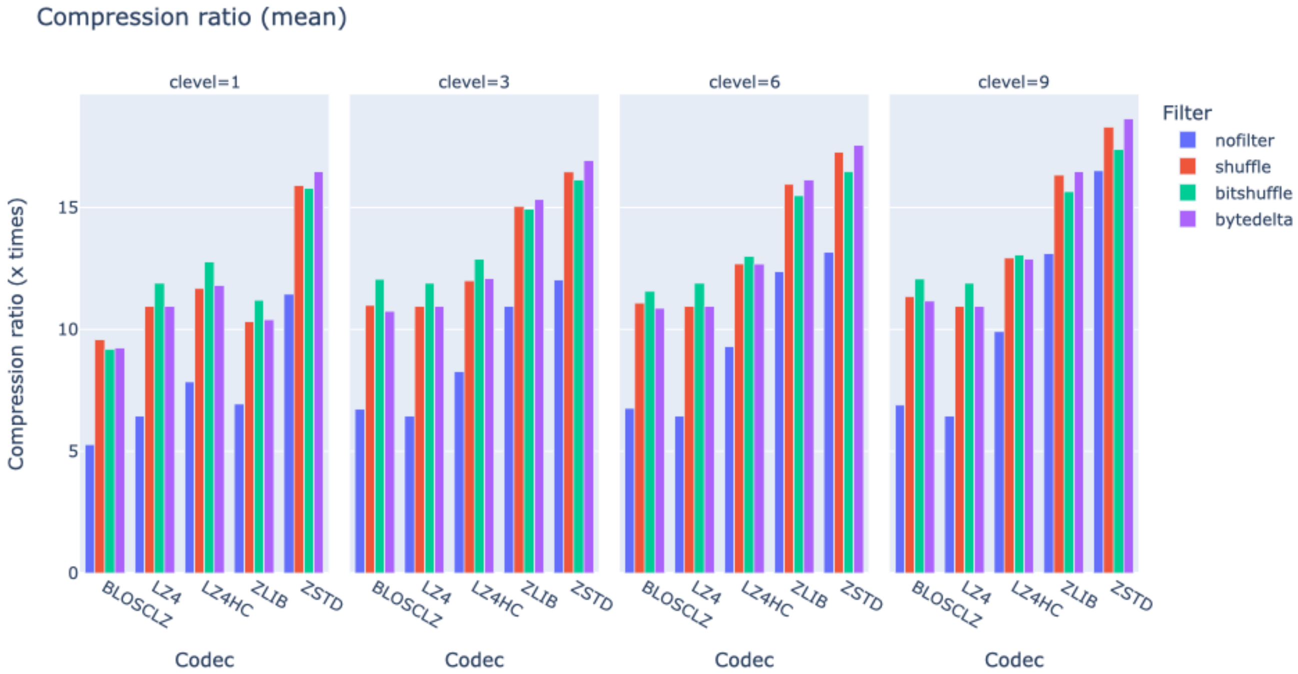
Trailer:
Var Length
Metalayers
(up to 2 GB)

vlmeta1

vlmeta2

vlmeta3

Different Codecs and Filters



How to predict the best combination?
https://ironarray.io/btune

Blosc2: Compute Engine

Compute with your big compressed arrays, fast!

Blosc2 Compute Engine: Computing With Compressed Data, Transparently

For optimal speed, it's crucial to **understand** and **utilize** modern CPU capabilities:

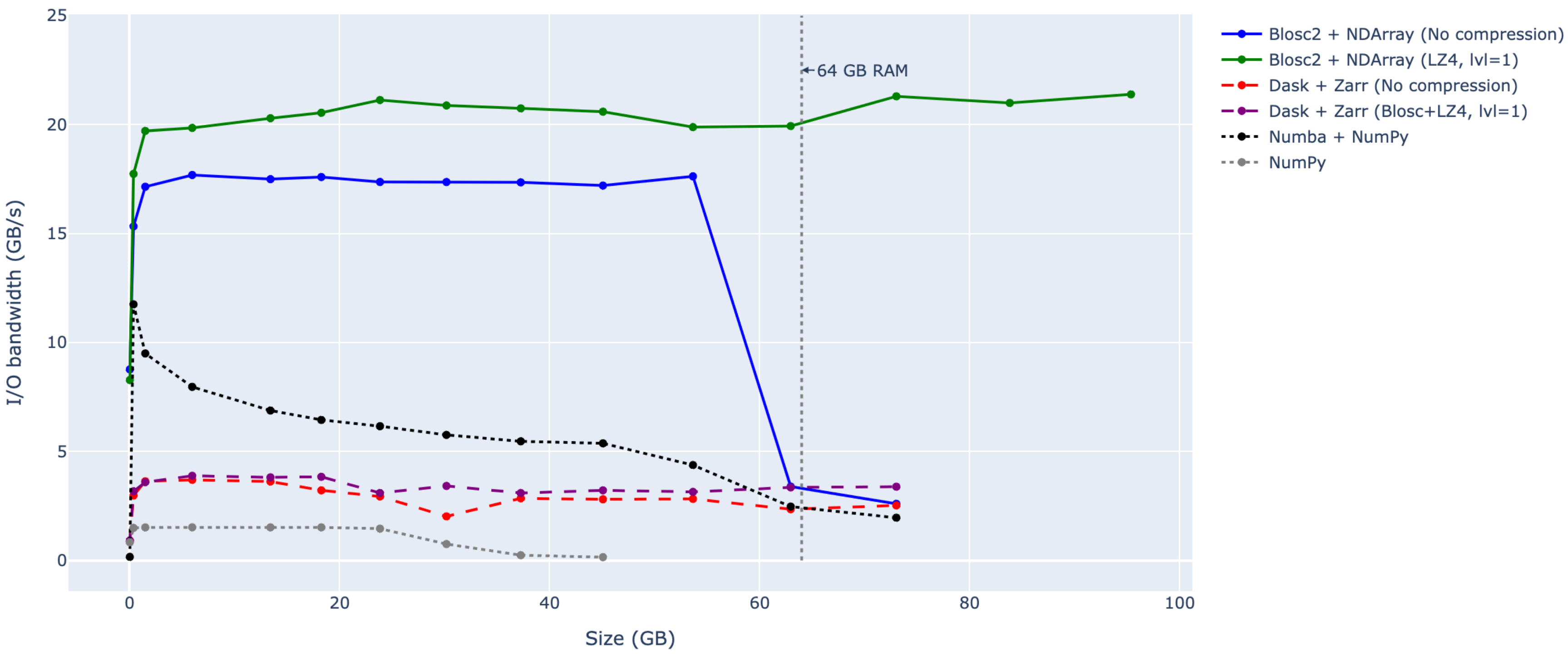
- Multicore processing
- SIMD
- Cache hierarchies

```
N = 1000_000
a = blosc2.linspace(0, 1, N * N, shape=(N, N))
b = blosc2.linspace(1, 2, N * N, shape=(N, N))
c = blosc2.linspace(0, 1, N, shape=(N,)) # 1D; broadcasting supported
# Blosc2 NDArrays override NumPy's universal functions (ufuncs)
out = np.sum(((a ** 3 + np.sin(a * 2)) < c) & (b > 0), axis=1)
```

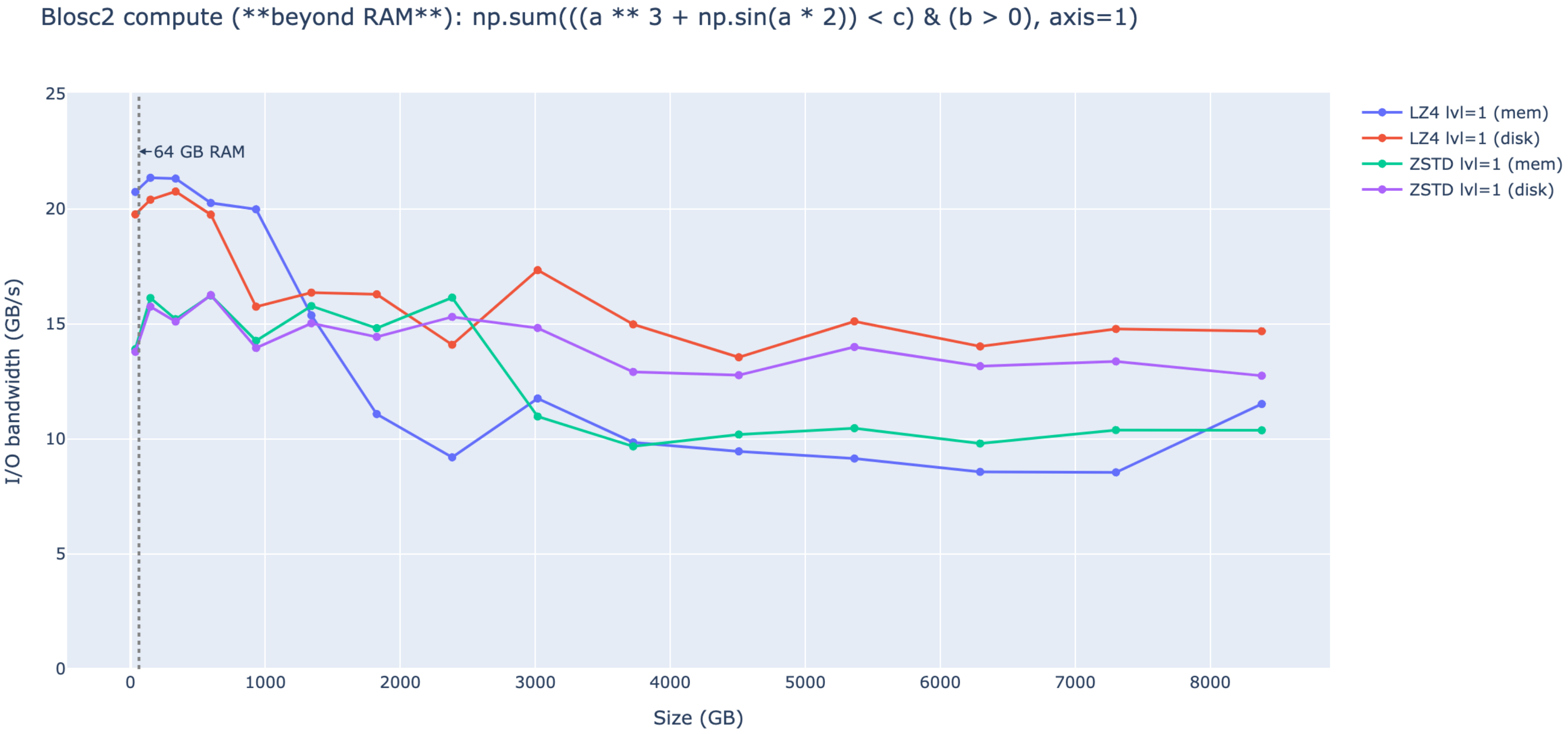
https://www.blosc.org/python-blosc2/getting_started/overview.html#computing-with-ndarrays

Efficient Compressed Computing

Blosc2 vs others; compute: np.sum(((a ** 3 + np.sin(a * 2)) < c) & (b > 0), axis=1)



Going Bigger: Computing Beyond RAM



https://ironarray.io/blog/compute-bigger

Blosc2: TreeStore

Endow a structure to your data, effortlessly

Blog: https://www.blosc.org/posts/new-treestore-blosc2/

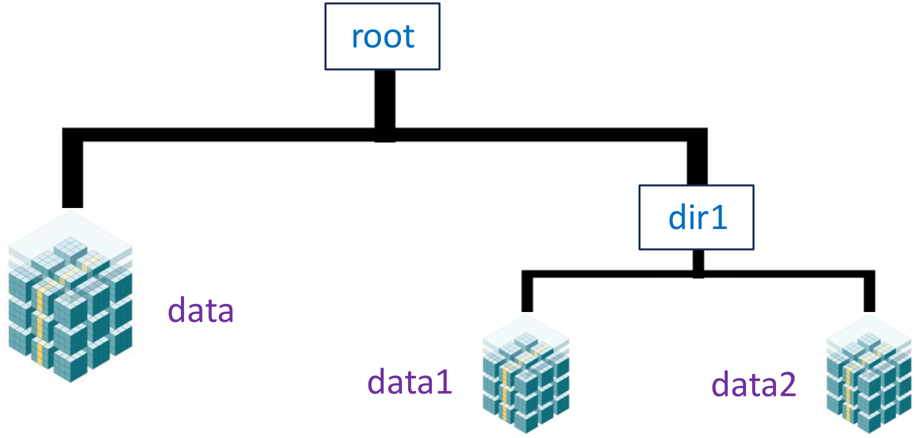
Hierarchical Structures

blosc2.TreeStore

New in Python-Blosc2 3.7.0

Currently in beta stage.

Sprinting tomorrow, join us!



root

dir1

data

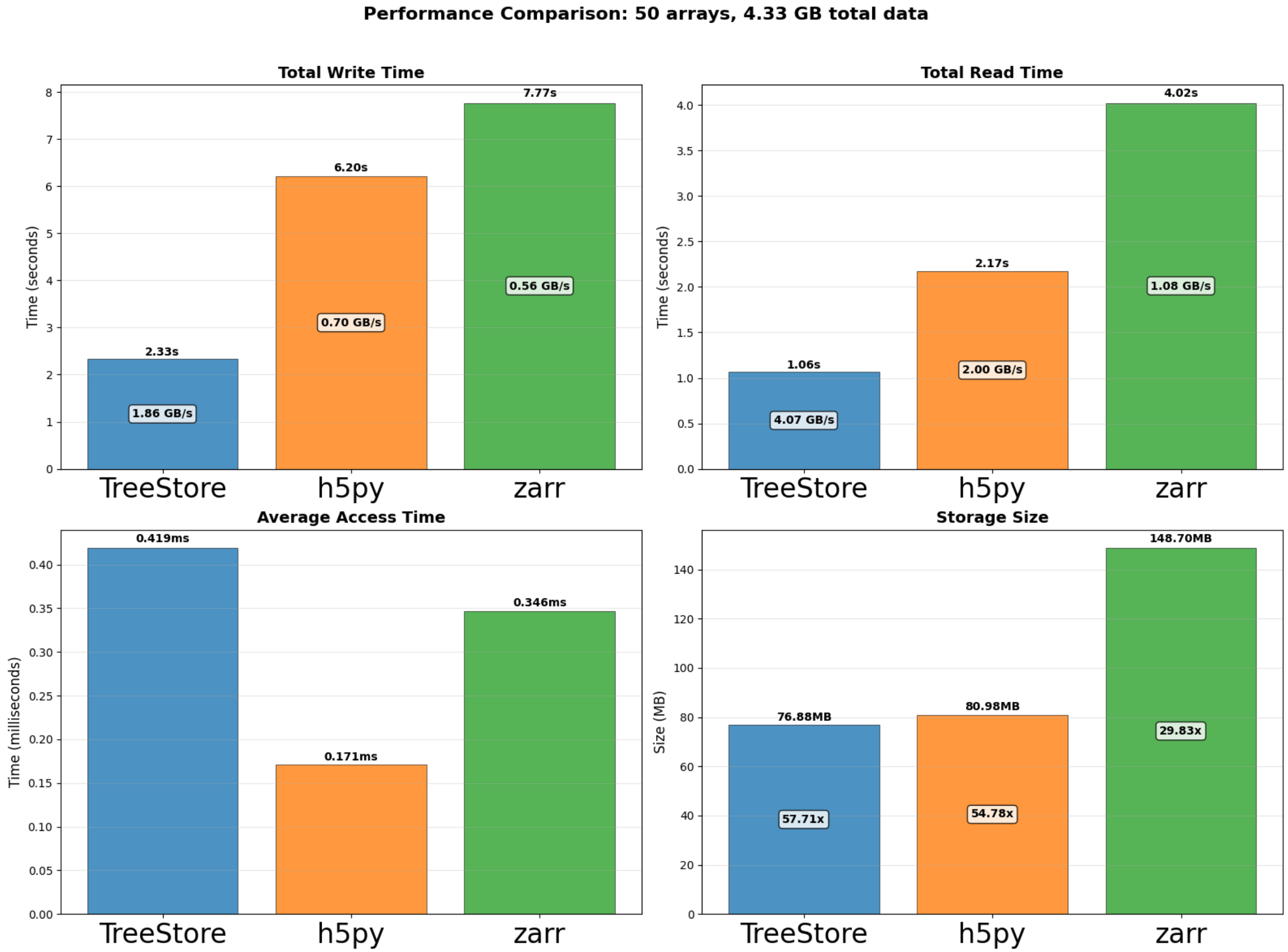
data1

data2

```
with blosc2.TreeStore("example_tree.b2z", mode="w") as tstore:
    tstore["/data"] = np.array([1, 2, 3])  # numpy array
    tstore["/dir1/data1"] = blosc2.ones(1e6, shape=(100, 100, 100)) # blosc2 array
    tstore["/dir1/data2"] = blosc2.linspace(0, 1, 1e6, shape=(1000, 1000))
    tstore.vlmeta["author"] = "blosc2”  # metadata (persists with the store)

with blosc2.TreeStore("example_tree.b2z", mode="r") as tstore2:
    print("/dir1/data2:", tstore2["/dir1/data2"][:])
```

Bench on MacBook Air, M2 CPU, 8 threads (default)



Performance Comparison: 50 arrays, 4.33 GB total data

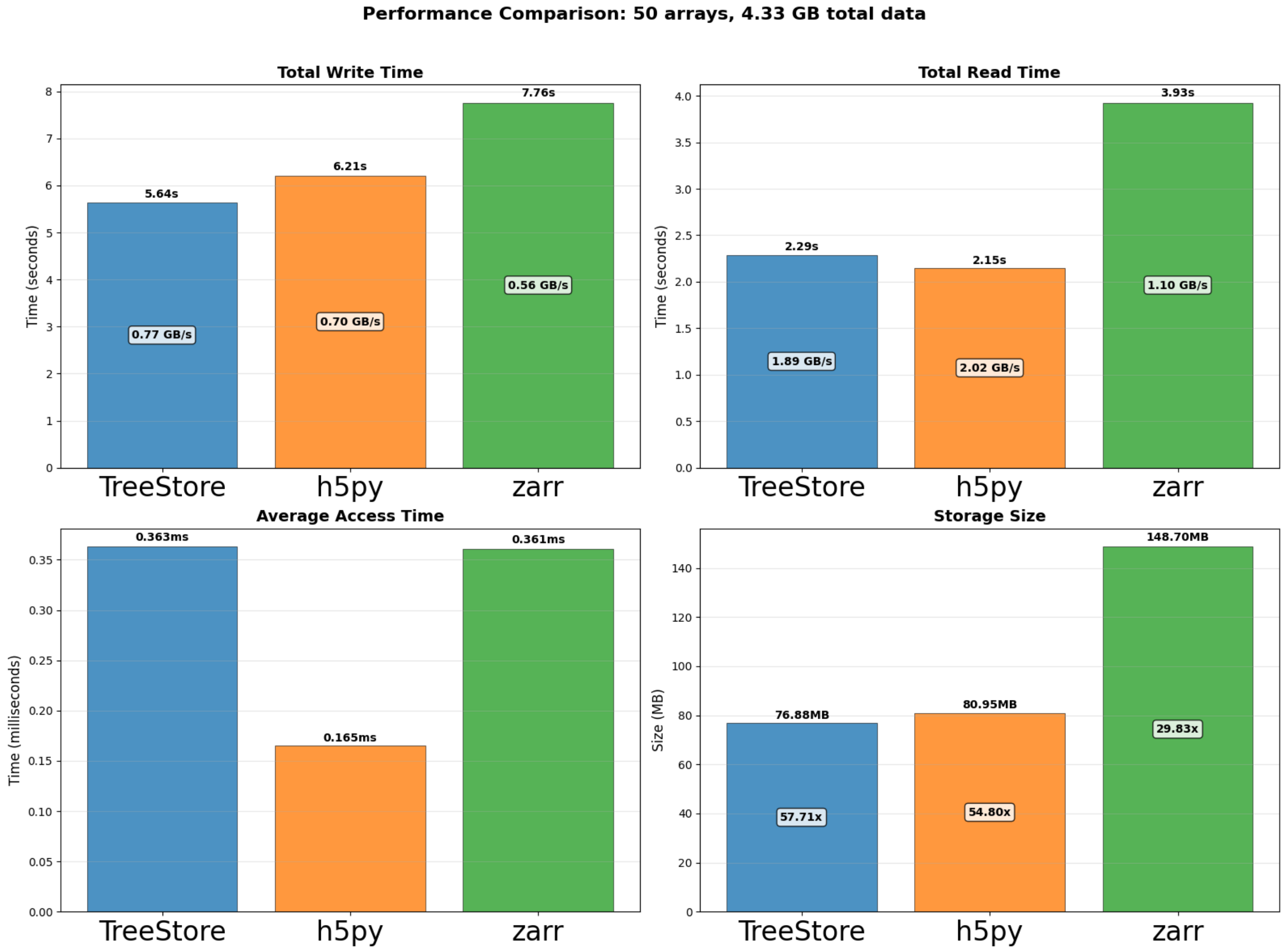
Total Write Time

Total Read Time

Average Access Time

Storage Size

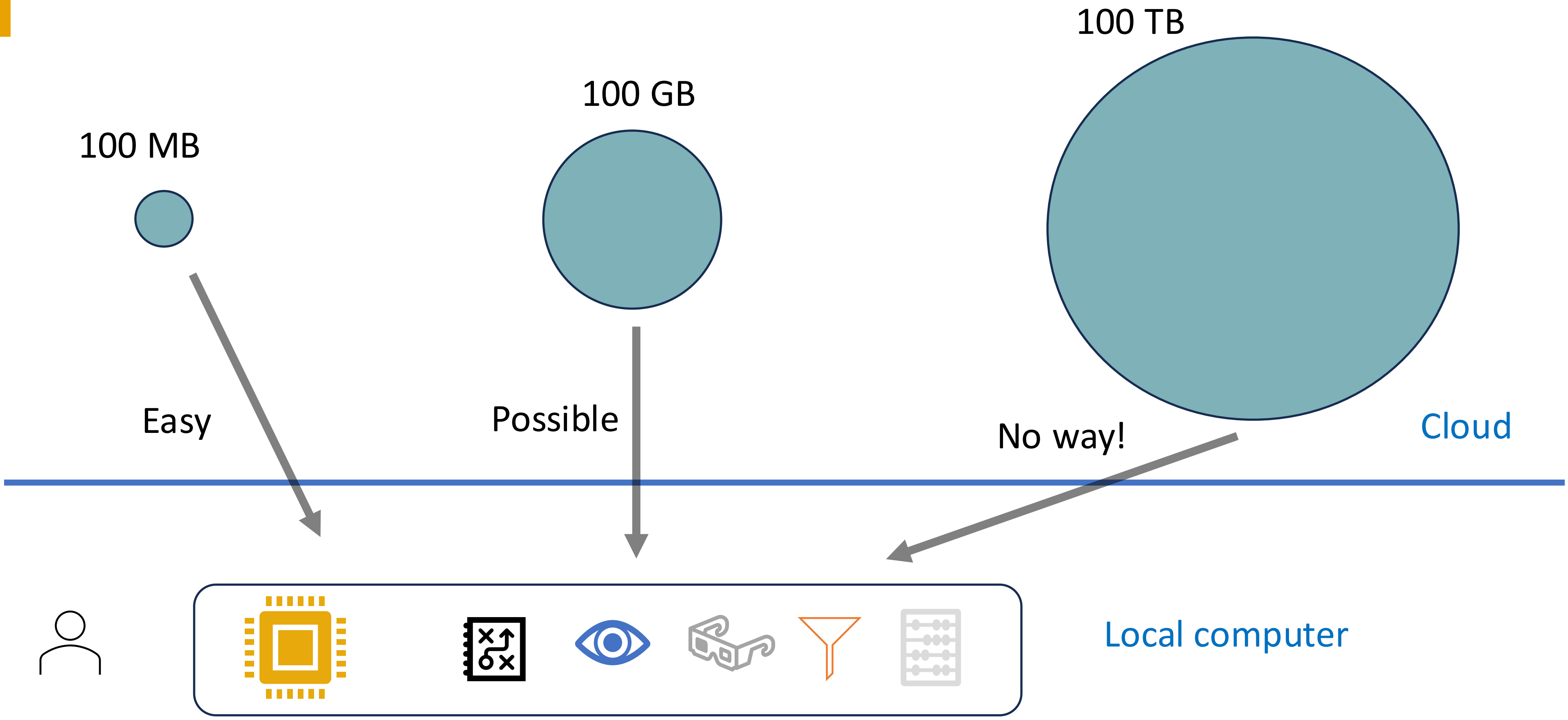
Bench on MacBook Air, M2 CPU, 1 thread (forced)



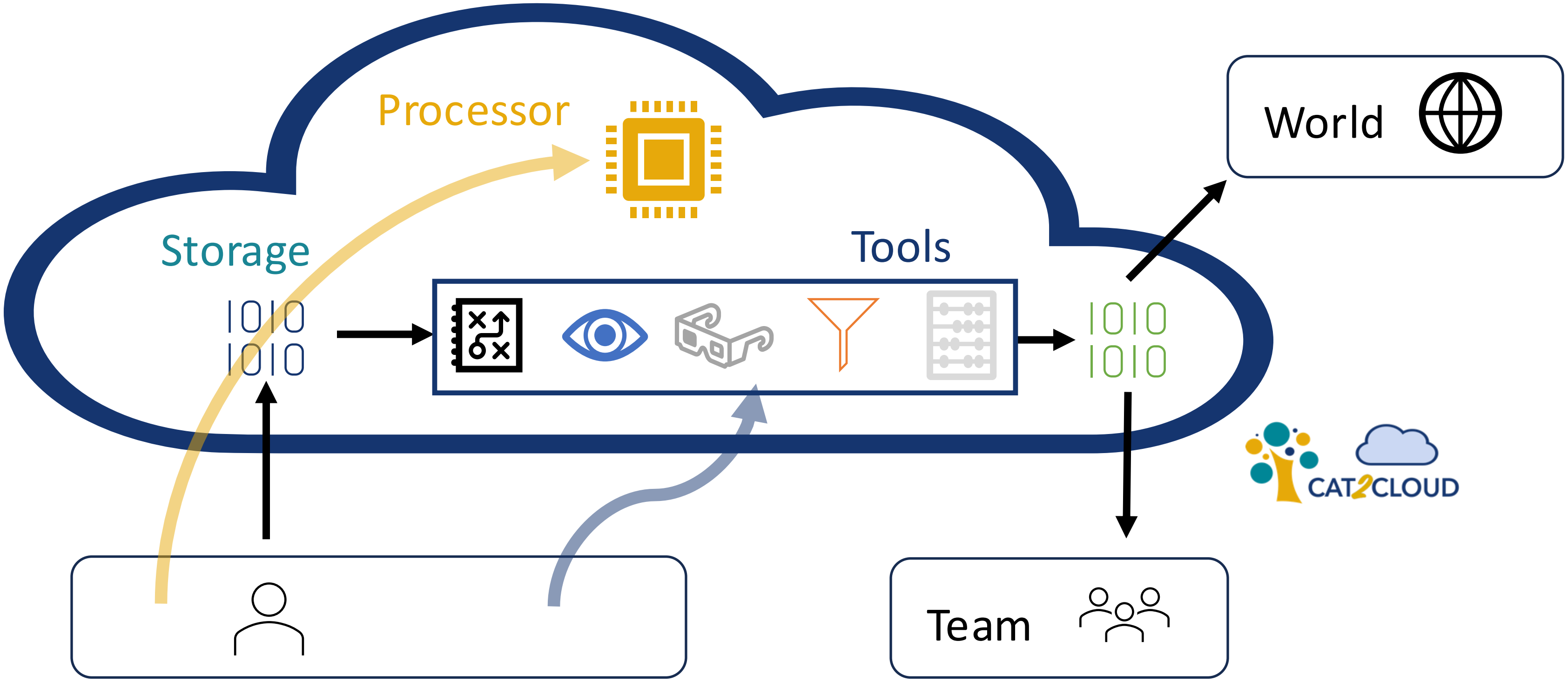
CATERVA2

Bring Computation Closer To Where Data Is Stored

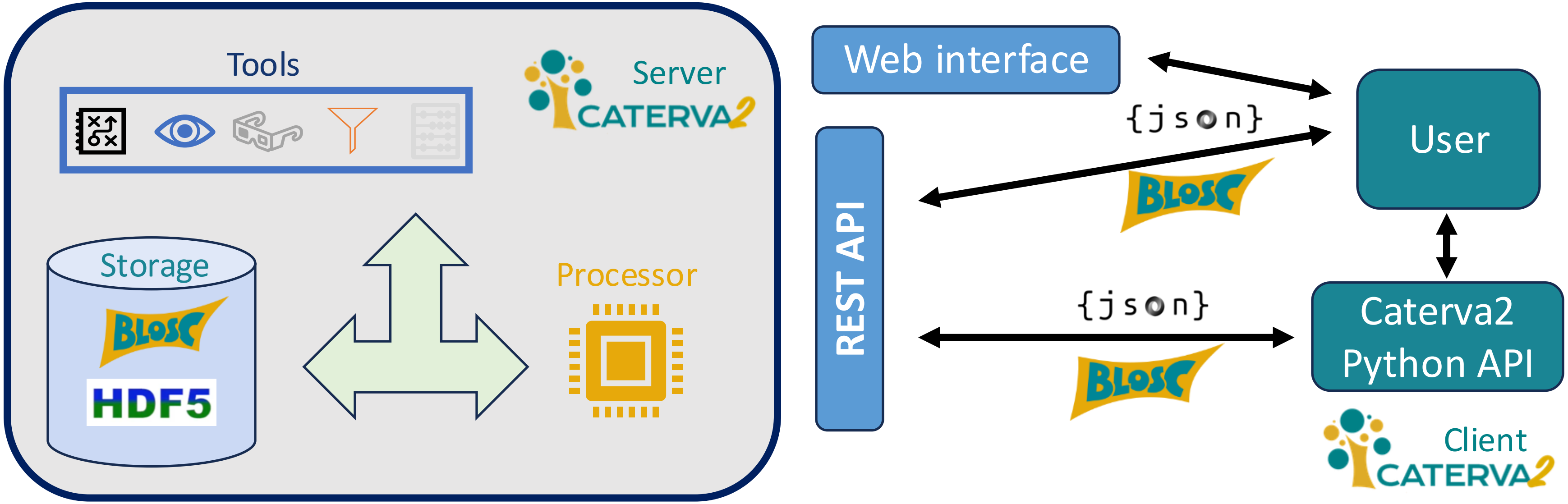
Data Is Affected By Physical Laws!



Computation Needs To Be Closer To Were Data Is Stored



Serving data through Internet



https://ironarray.io/caterva2

https://github.com/ironArray/Caterva2

Conclusion

Blosc2 And Its Ecosystem: Tooling For Modern Computing

Python-Blosc2:

- Efficient compression and adaptability: Compress better
- Computation intertwined with compression: Compute bigger

Caterva2:

- Bring these to the cloud, with easy: Share faster

Don’t worry (too much) about performance,
use Blosc2 ecosystem, and do your thing!

Thanks to Donors & Contracts!

HUAWEI

[[ ]] ironArray

NUMFOCUS
OPEN CODE = BETTER SCIENCE

python SOFTWARE FOUNDATION

LEAPS INNOVATION

ESRF
The European Synchrotron

Google

Jeff Hammerbacher

Thanks! Questions?

Blosc

ironArray

blosc@blosc.org
contact@ironarray.io

Compress Better, Compute Bigger, Share Faster