Mastering Large NDArray Handling with Blosc2 and Caterva2

Francesc Alted / @FrancescAlted@masto.social

The Blosc Development Team / @Blosc2@fosstodon.org

CEO ironArray / francesc@ironArray.io

PyData Global 2024
December 3th 2024

Agenda

- Intro
- **NDArray**: Blosc2's N-dim data container
- **LazyArray:** Computing expressions and UDFs
- **Caterva2:** Access to Blosc2 data from network
- Conclusions

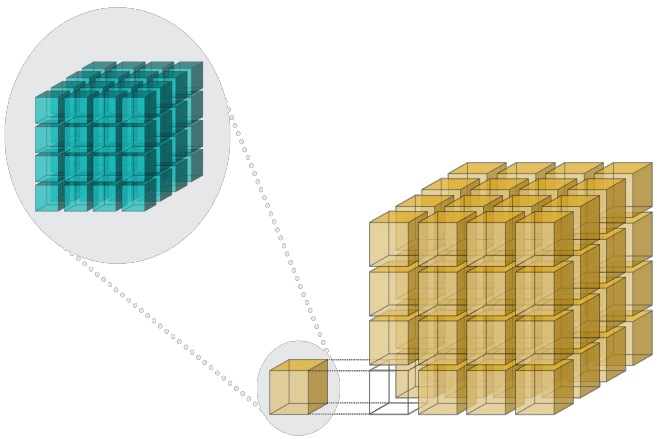
A Climate Warning

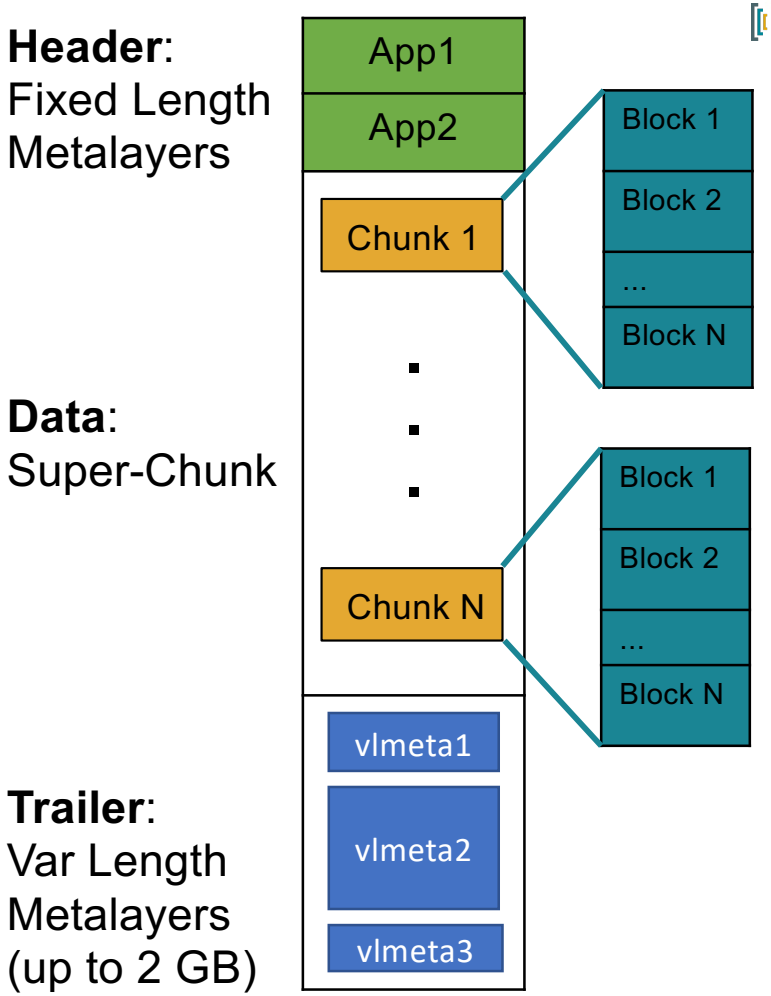
- Catastrophic rain in València area a month ago
- More that 200 deaths
- Tens of thousands Meuros in loses
- Global Warming is real!
- **Act and reduce you carbon footprint!**

Intro

What is Blosc2?

- ✓ Adds 63-bit containers.
- ✓ Metalayers for adding info for apps and users.
- ✓ Multidimensional blocks and chunks.





Who is ironArray SLU?

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

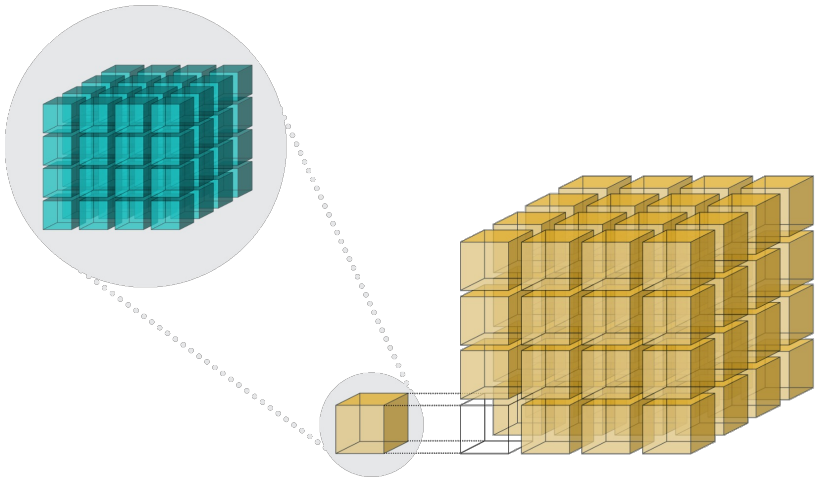




ironArray

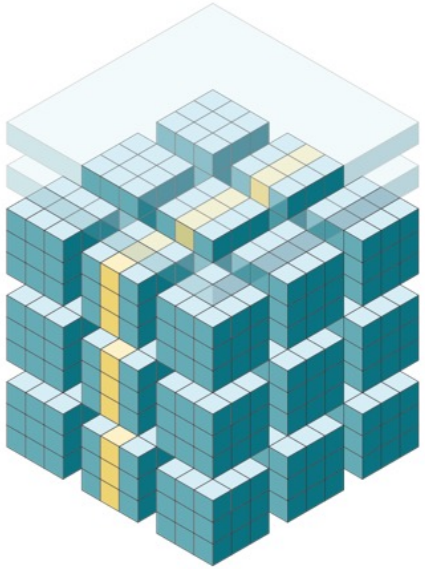
https://ironarray.io

NDArray: N-Dim and compressed data

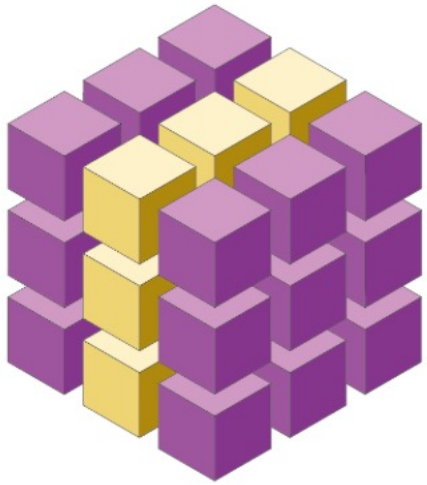


Leveraging the second partition in Blosc2 NDim

Much more selective and hence, faster queries!

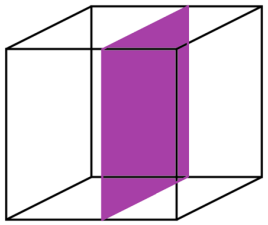


Blosc2 NDim

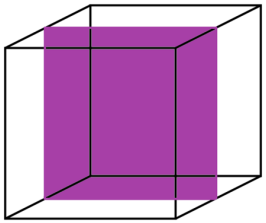


HDF5 / Zarr / others

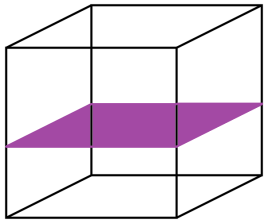
Reading orthogonal slices



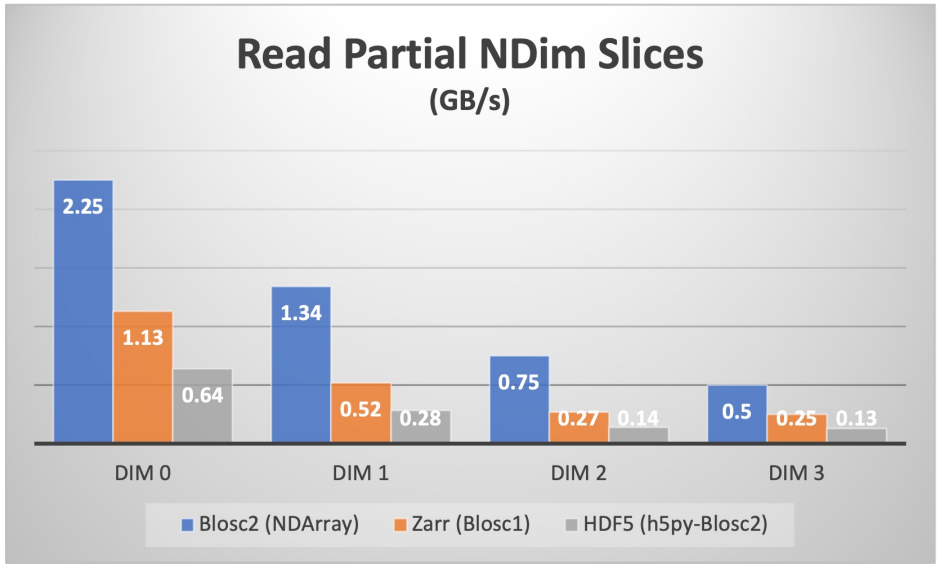
constant dim0



constant dim1



constant dim2



Faster slicing due to higher data selectivity in double partitioning

Hands-on time

Do these actions in command line:

- `git clone` https://github.com/Blosc/Python-Blosc2-3.0-tutorial
- `conda create -n blosc2-tutorial python=3.12`
- `conda activate blosc2-tutorial`
- `pip install -r requirements.txt`
- `jupyter lab`

And let’s start with the first tutorial.

LazyArray: Computing expressions and UDFs

Computing expressions

You can perform a rich variety of mathematical expressions:

```
a = blosc2.arange(10)
la = sin(a) **3 + 1
```

The result (la) is an object that follows the LazyArray interface.

This allows to operate with your NDArray objects on a lazy manner.

User Defined Functions

If expressions are not flexible enough, you can define your own function and use it for doing computations with arbitrary inputs. E.g.

```
def myudf(inputs_tuple, output, offset):
    x, y = inputs_tuple
    output[:] = x**3 + np.sin(y) + 1

# a and b can be NDArrays or NumPy arrays
la = blosc2.lazyudf(myudf, (a, b), a.dtype)
```

Hands-on time

- Let’s continue with tutorial 2 (expressions)
- And then tutorial 3 (User Defined Functions)

Caterva2: On-demand access to remote Blosc2 datasets

Caterva2 highlights

- Share your Blosc2 data (and any kind of data actually) through the network.
  - Use a web interface: https://ironarray.io/caterva2-doc/tutorials/web-client.html
  - Command line: https://ironarray.io/caterva2-doc/tutorials/cli.html
  - High-level API: https://ironarray.io/caterva2-doc/tutorials/API.html

We provide **cat2cloud**, a cloud service for Caterva2: https://ironarray.io/cat2cloud

Hands-on time

- Go to cat2.cloud/demo and try the interface with me.
- Three groups:
  - @personal: only you can see or remove files here
  - @team: all your team can see or remove files
  - @public: the world can see everything (not remove)
- Try upload anything, from .b2nd files to .png, .pdf or .md, and visualize them.

Conclusion

Python-Blosc2 3.0 is ready for work!

With recently published 3.0.0 rc2 release, you can:

- Work with **large NDArrays**, be in-memory, disk or on the network
- **Compute** arbitrarily complex expressions (including reductions!) on a lazy manner
- Support for **User Defined Functions**
- With **Caterva2**, you can share your Blosc2 data in internet with easy and efficiency.

Blosc2: a highly efficient and flexible tool for
compressing and computing your data, your way

Thanks! Questions?





contact@ironarray.io

Compress Better, Compute Bigger, Share Faster