Migrating Astrophysical codes to exascale

ICT meeting - Bologna
30 Nov 2017

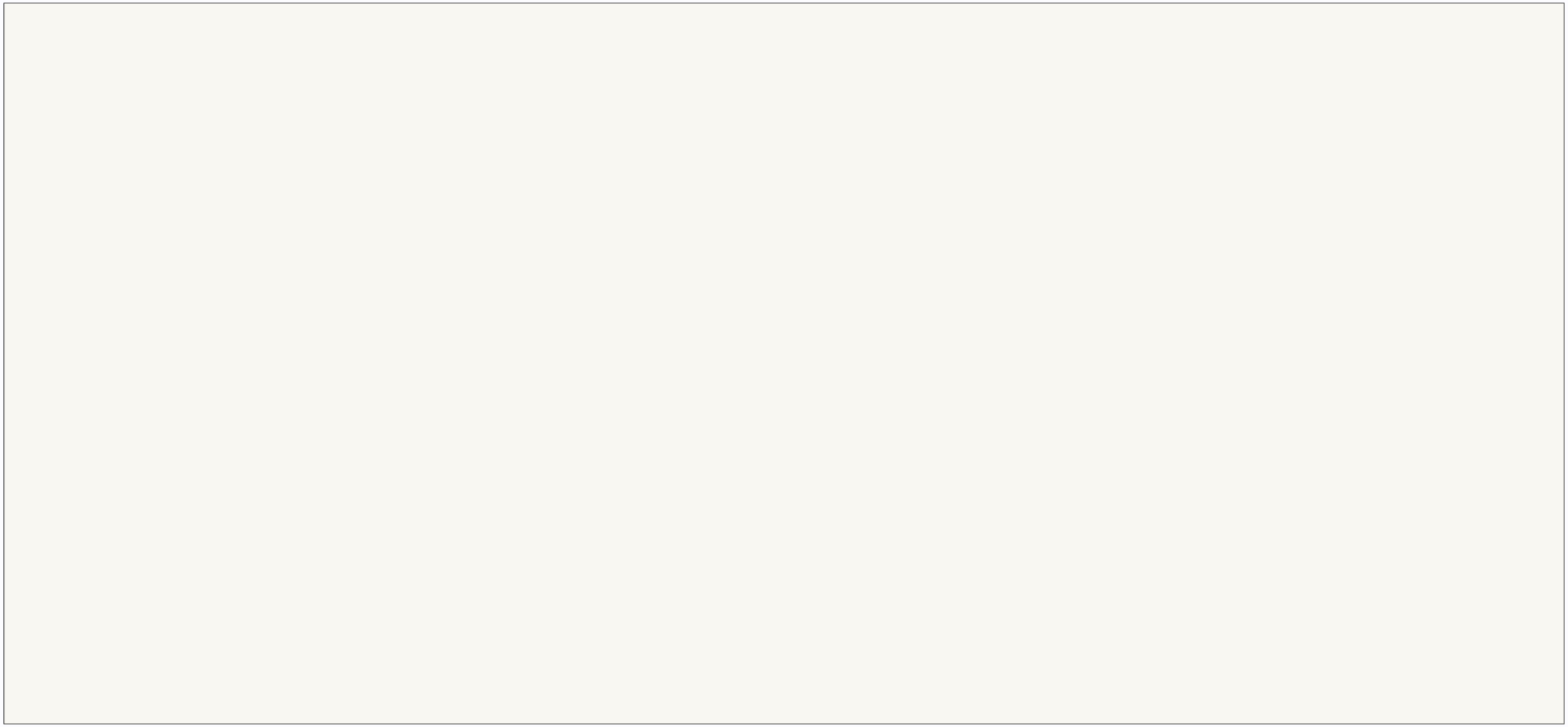
Luca Tornatore

Outline

- Exascale
- Project I : PINOCCHIO
- Project II : GADGET
- HPC Ecosystem @ HOTCAT

```
*** PBS Per-User Usage Report ***
User         Group          #Jobs        Job-Hours        CPU-Hours
_____________________________________________________________________
ltornatore   pinocchiox      4742         740.6236       90943.6678
```

Elements



➔ …just have a look at the live results..

EXASCALE – developers' point of view

POWER WALL

- **Code optimization** becomes fundamental
- **Concurrency** programming
- Cores' **specialization** must be exploited
- **Billion-way parallelism**

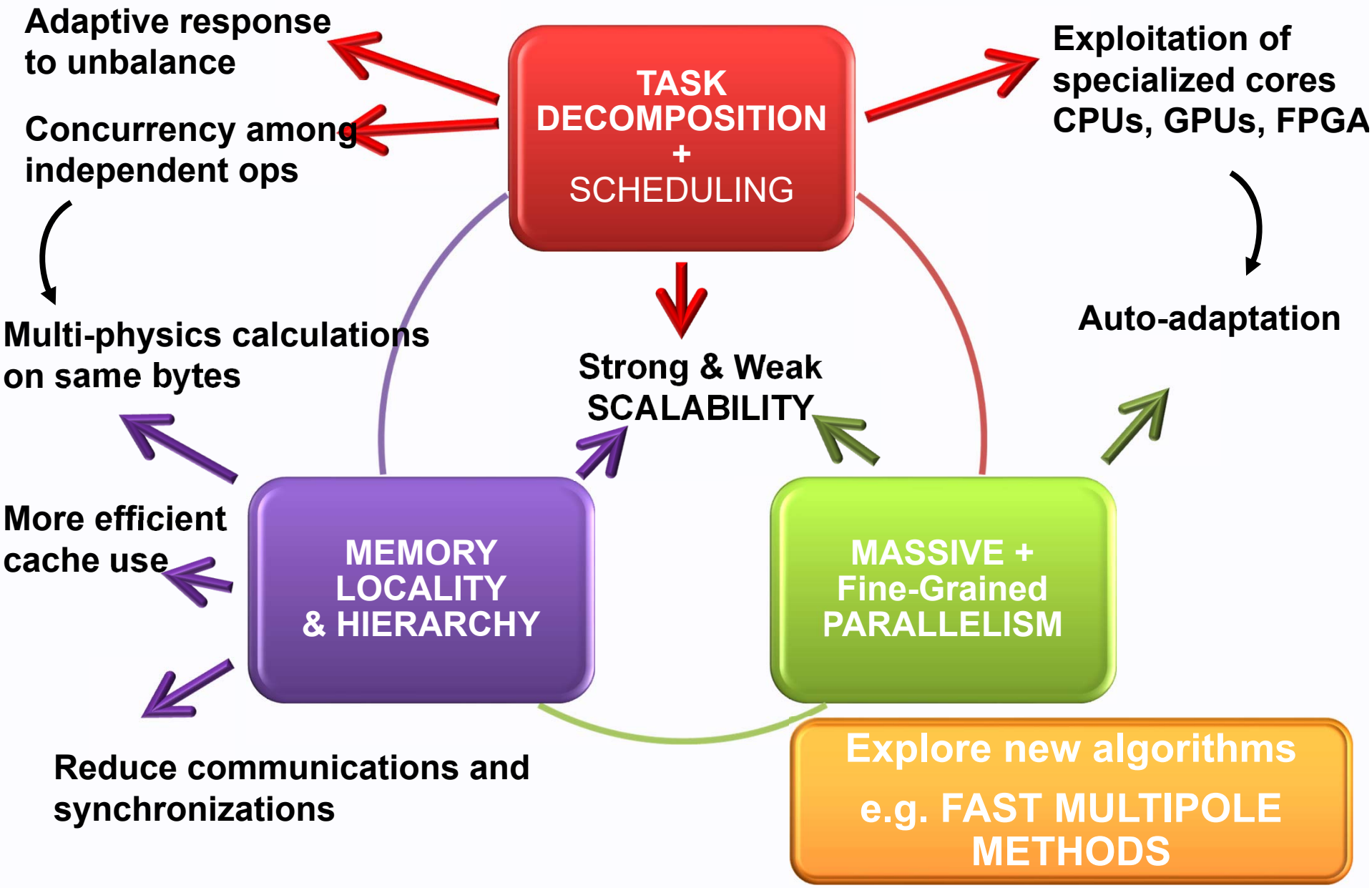
MEMORY WALL

- Small memory with high bw
- **Extreme** multi-level **NUMA** hierarchy
- Possible PGAS paradigm
- **Data locality/reuse by design** is mandatory

ILP -WALL

- **Concurrency** programming
- Careful **threadization**, do not rely on automatic pipelining

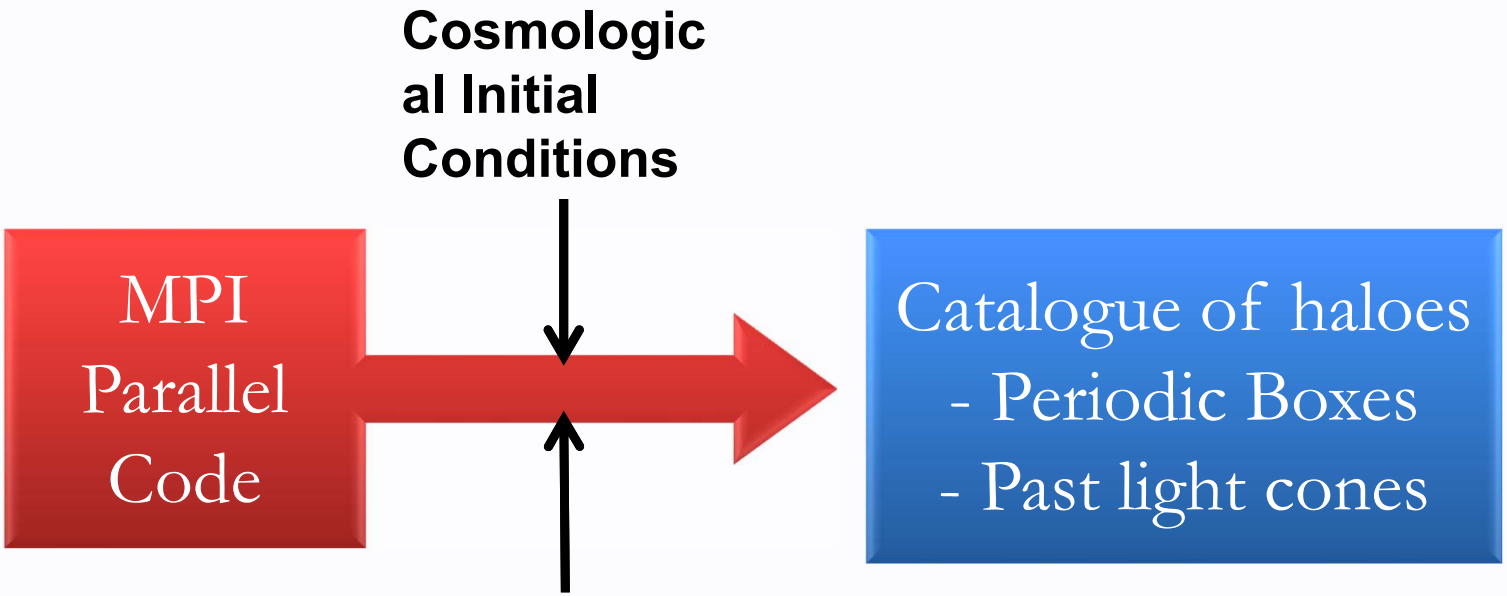
EXASCALE – developer's point of view



Outline

- Exascale – quick recap
- Project I : PINOCCHIO
- Project II : GADGET
- HPC Ecosystem @ HOTCAT

PINOCCHIO

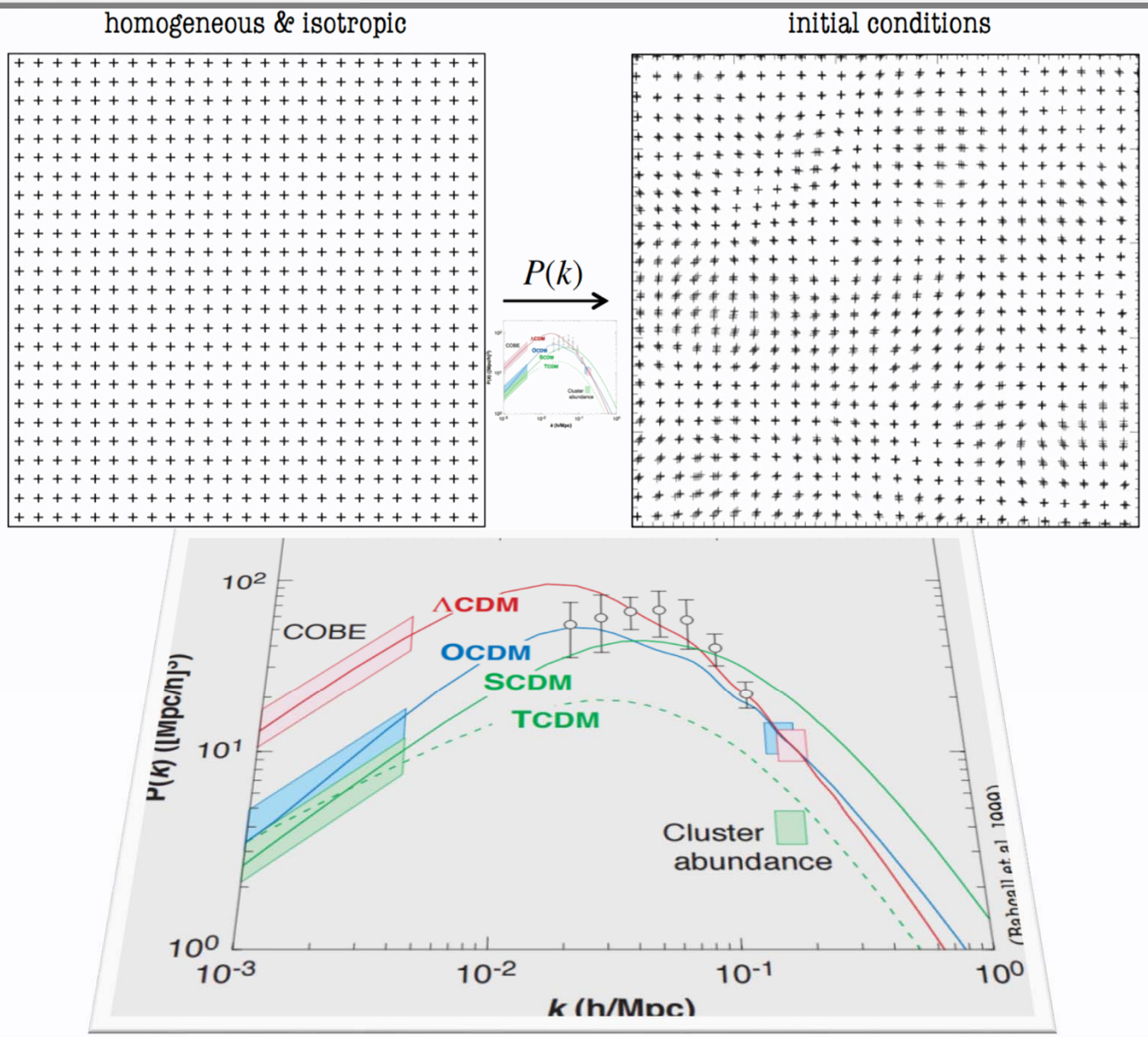


- **Lagrangian Perturbation Theory**
- **ellipsoidal collapse**
- **excursion sets theory.**

Used within the **Euclid** collaboration to produce large sets of dark matter halo catalogs to compute the covariance matrix of clustering measurements.

The code is distributed under GNU GPL 2 - On github, https://github.com/pigimonaco/Pinocchio

PINOCCHIO

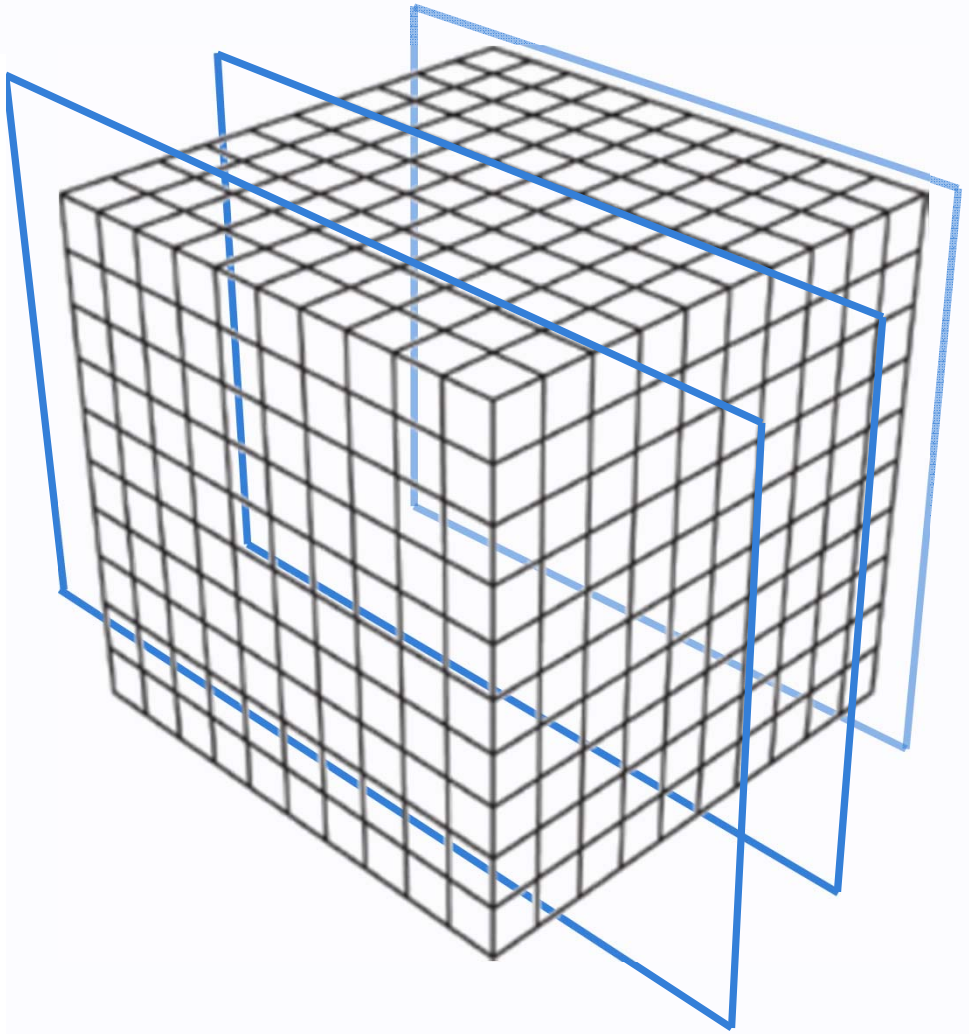

homogeneous & isotropic
initial conditions
P(k)
ΛCDM
COBE
OCDM
SCDM
TCDM
Cluster abundance
P(k) ([Mpc/h]³)
k (h/Mpc)
10²
10¹
10⁰
10⁻³
10⁻²
10⁻¹
10⁰
(Rahnall et al 1999)

PINOCCHIO

Using **FFTW** you are forced to parallelize the FFT by **slabs**.

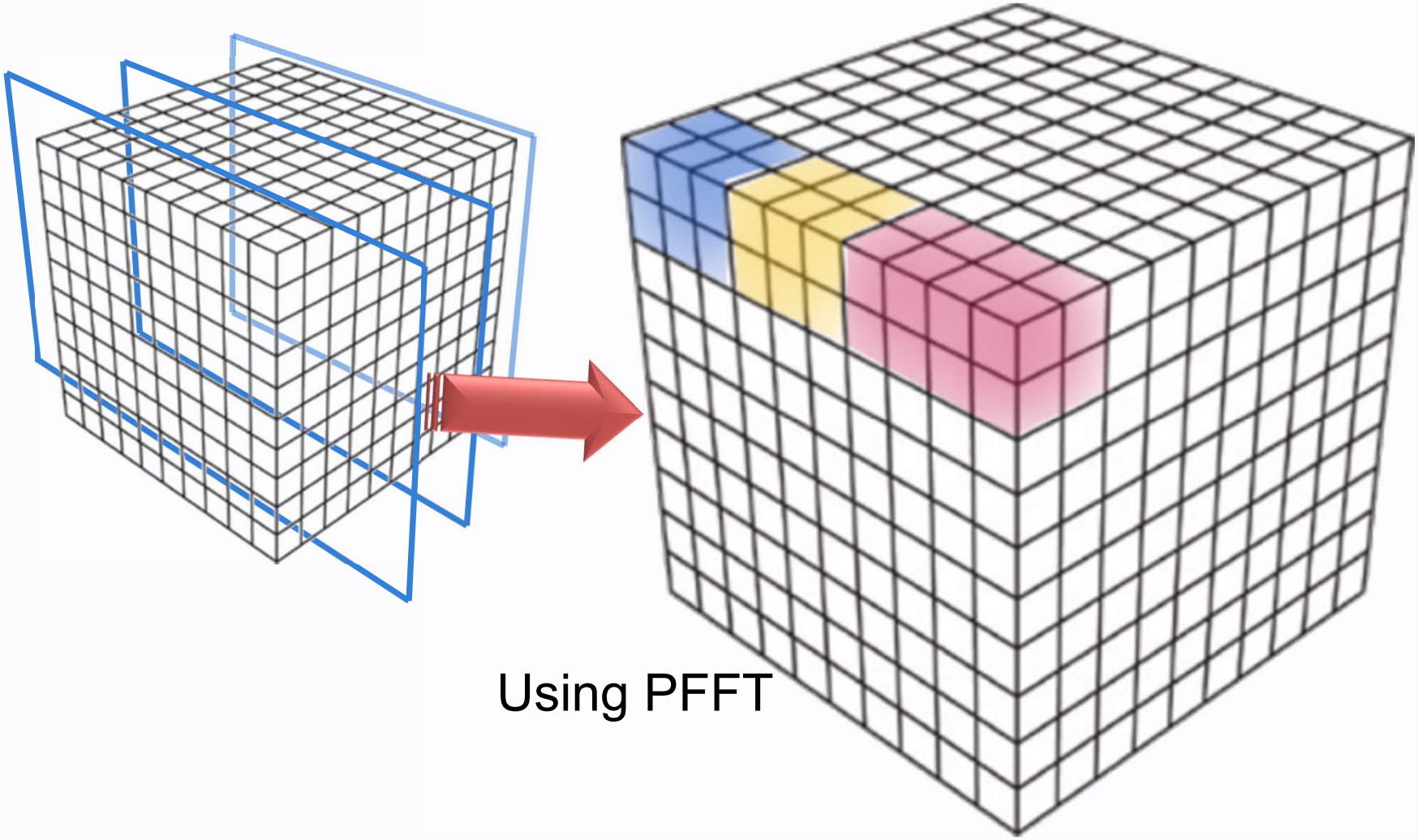
Then, **at maximum *N* of your N_T tasks are crunching $2\times N^2$ numbers**.

What **if *N* is insanely large** let's say ***exa-large:*** $2\text{-}3\times 10^4$?



PINOCCHIO

1) Move to a 3D decomposition for FFT

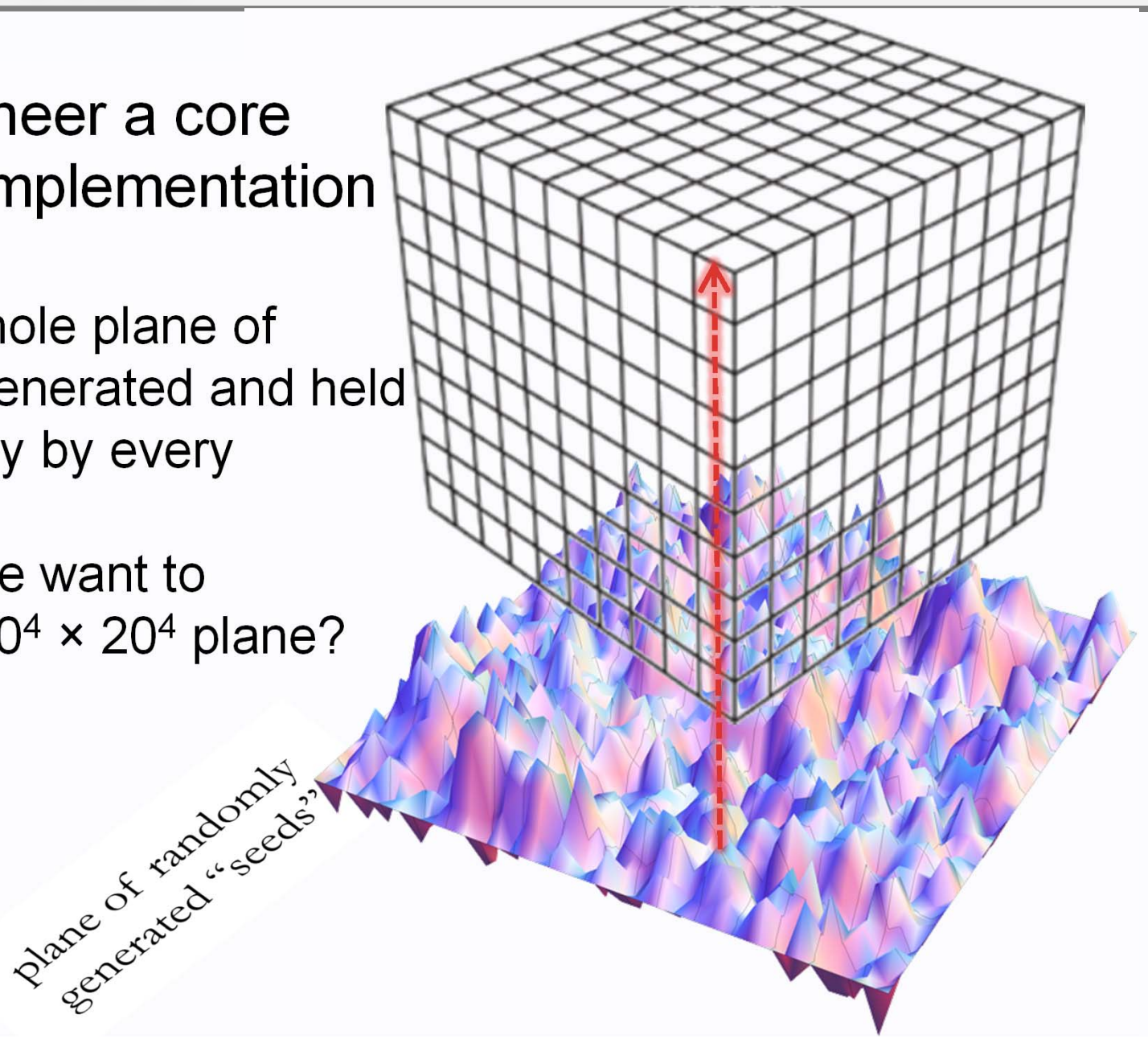


Using PFFT

PINOCCHIO

2) Re-engineer a core algorithm implementation

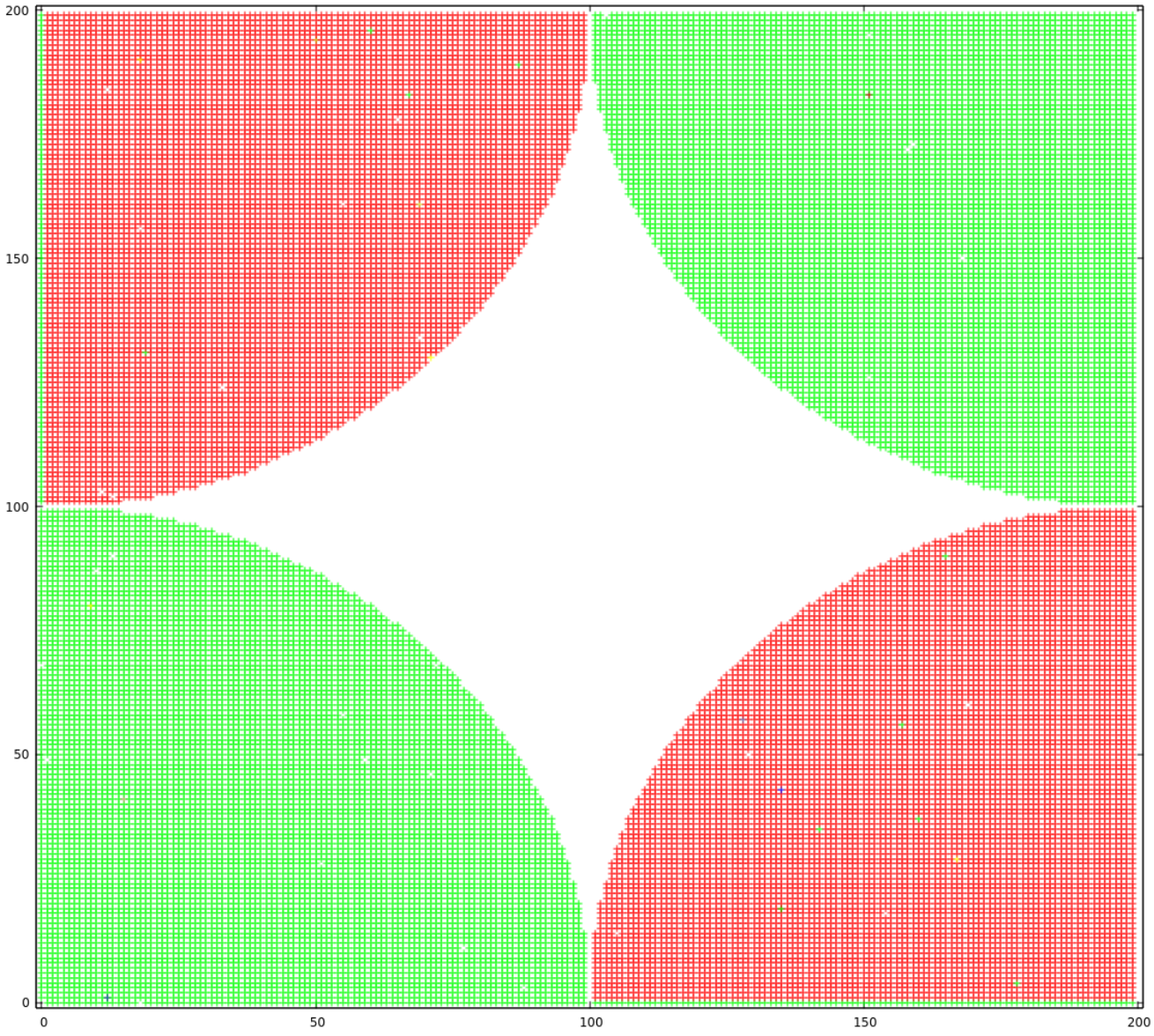
So far the whole plane of seeds was generated and held independently by every MPI tasks.
But what if we want to generate a $20^4 \times 20^4$ plane?



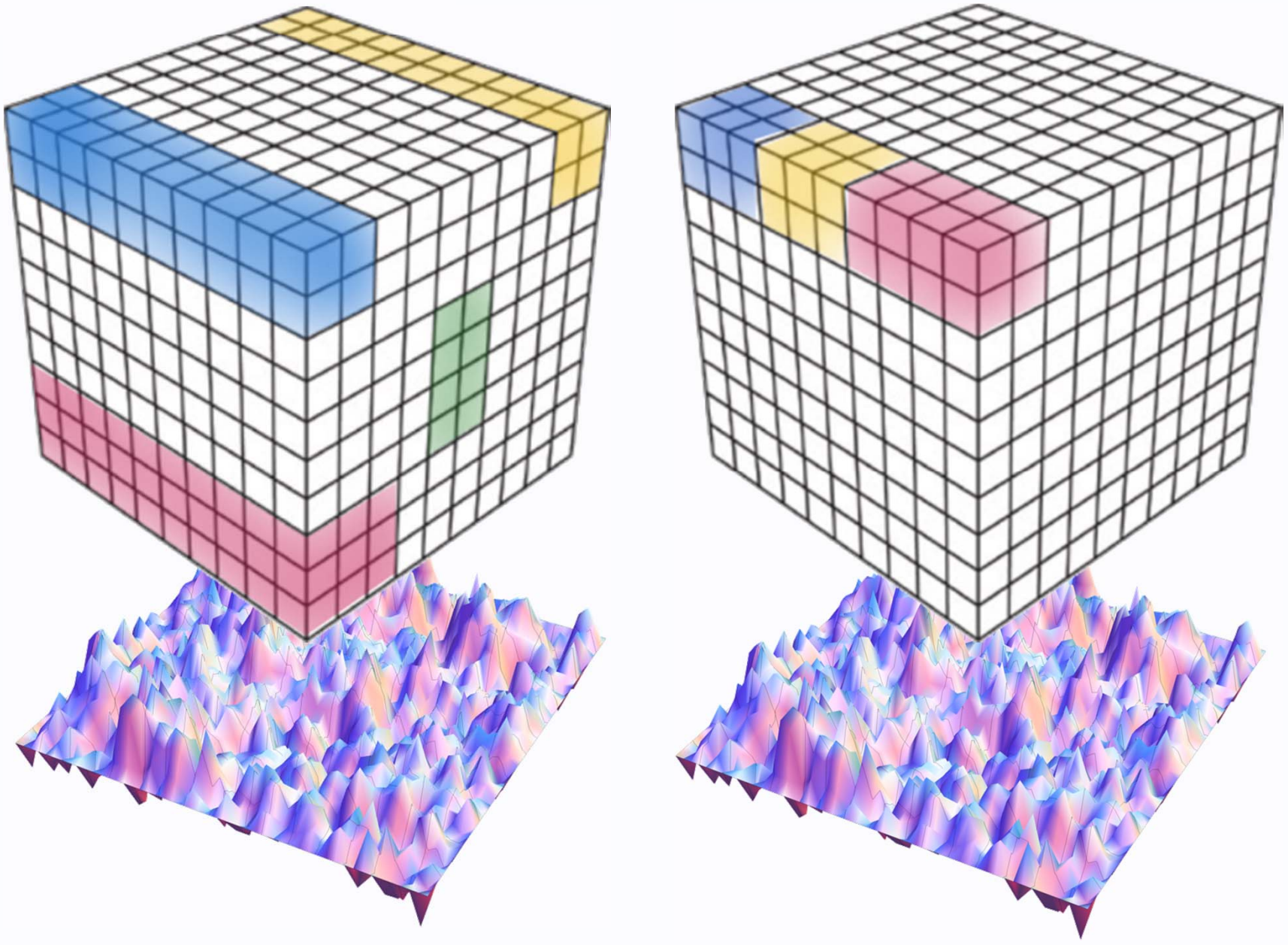
PINOCCHIO

Special symmetries on *k=0* plane

because the back-transformed data are a pure real field



PINOCCHIO

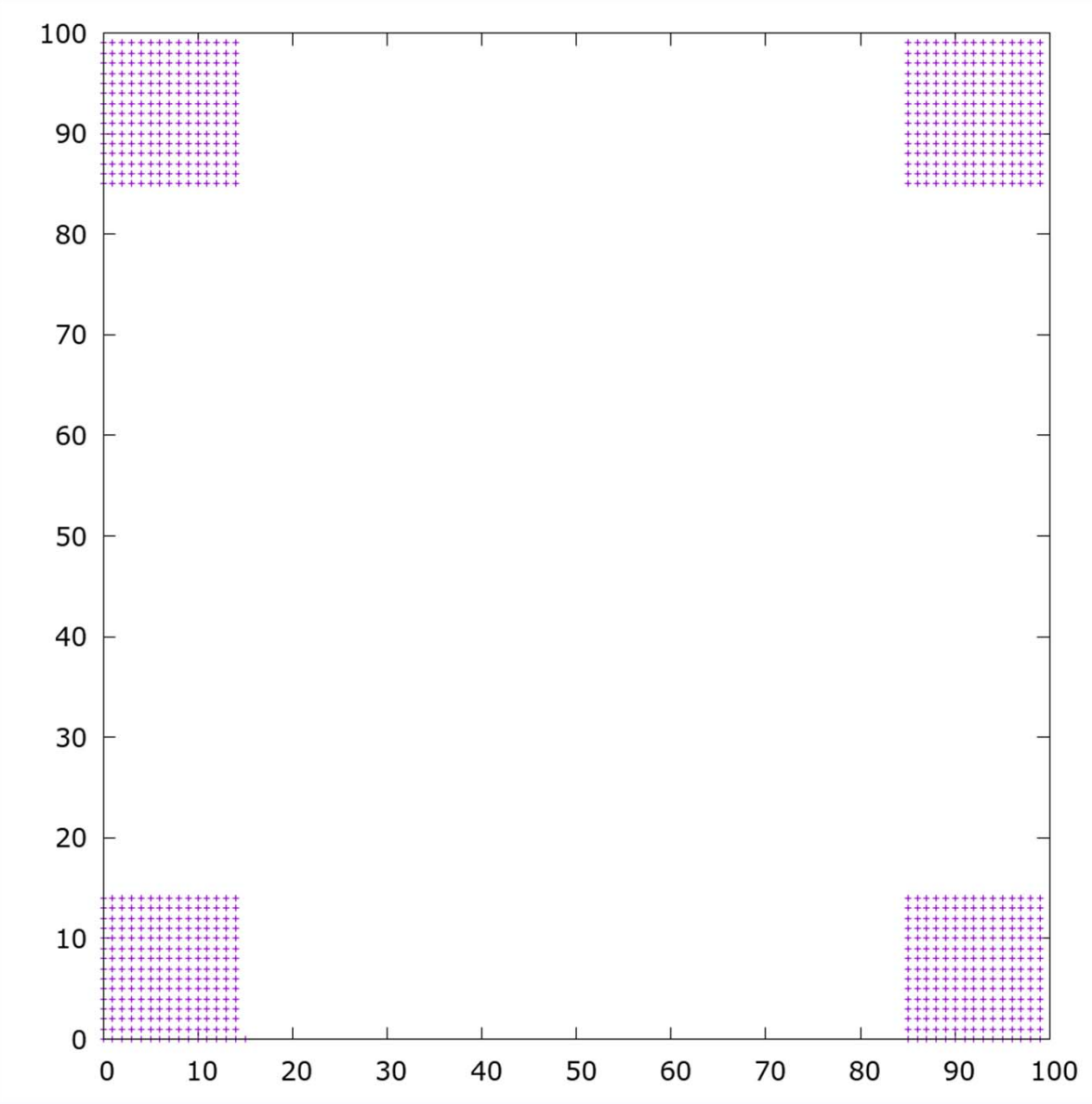


PINOCCHIO

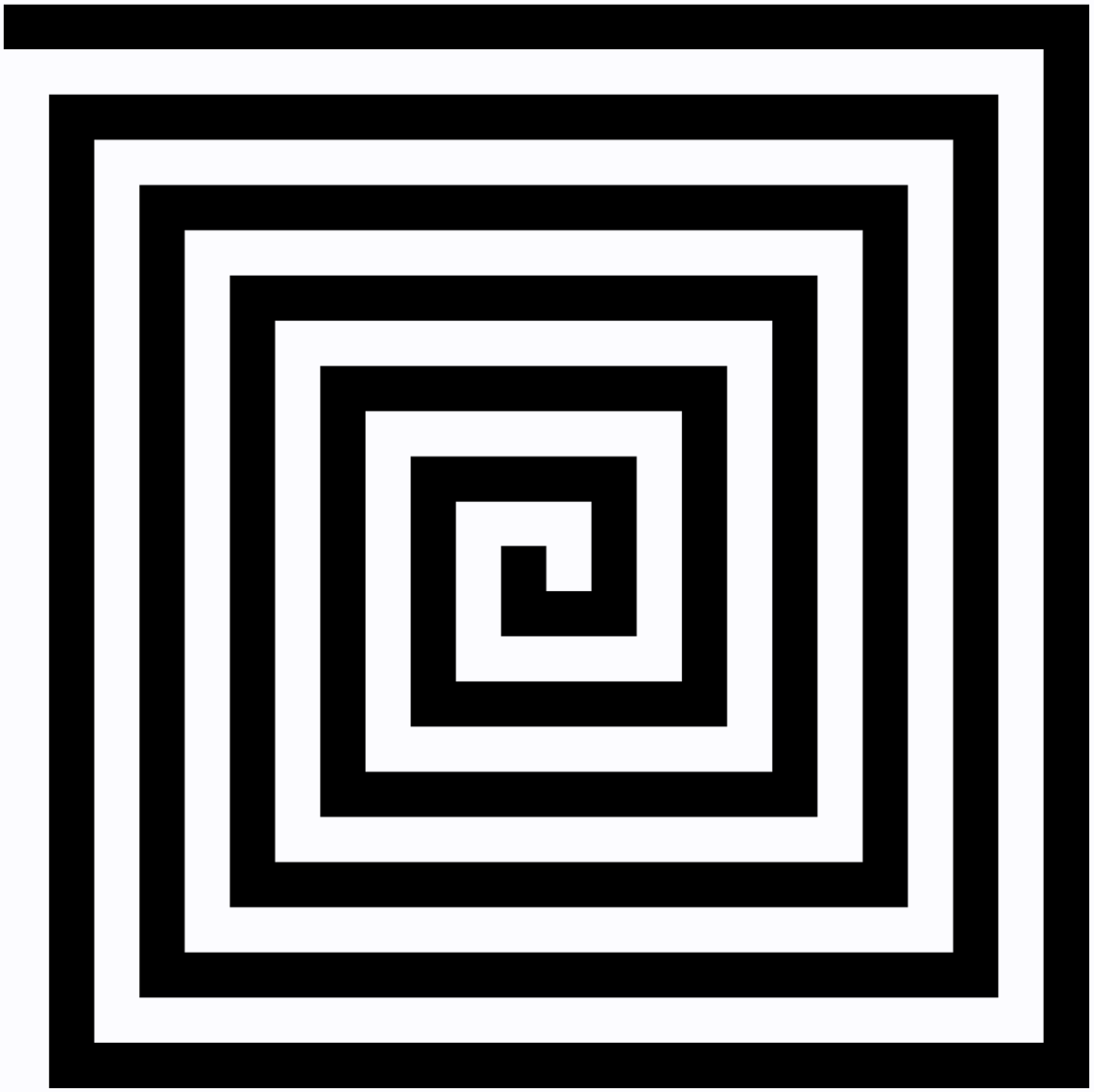
The seed plane must be generated with some special properties

→

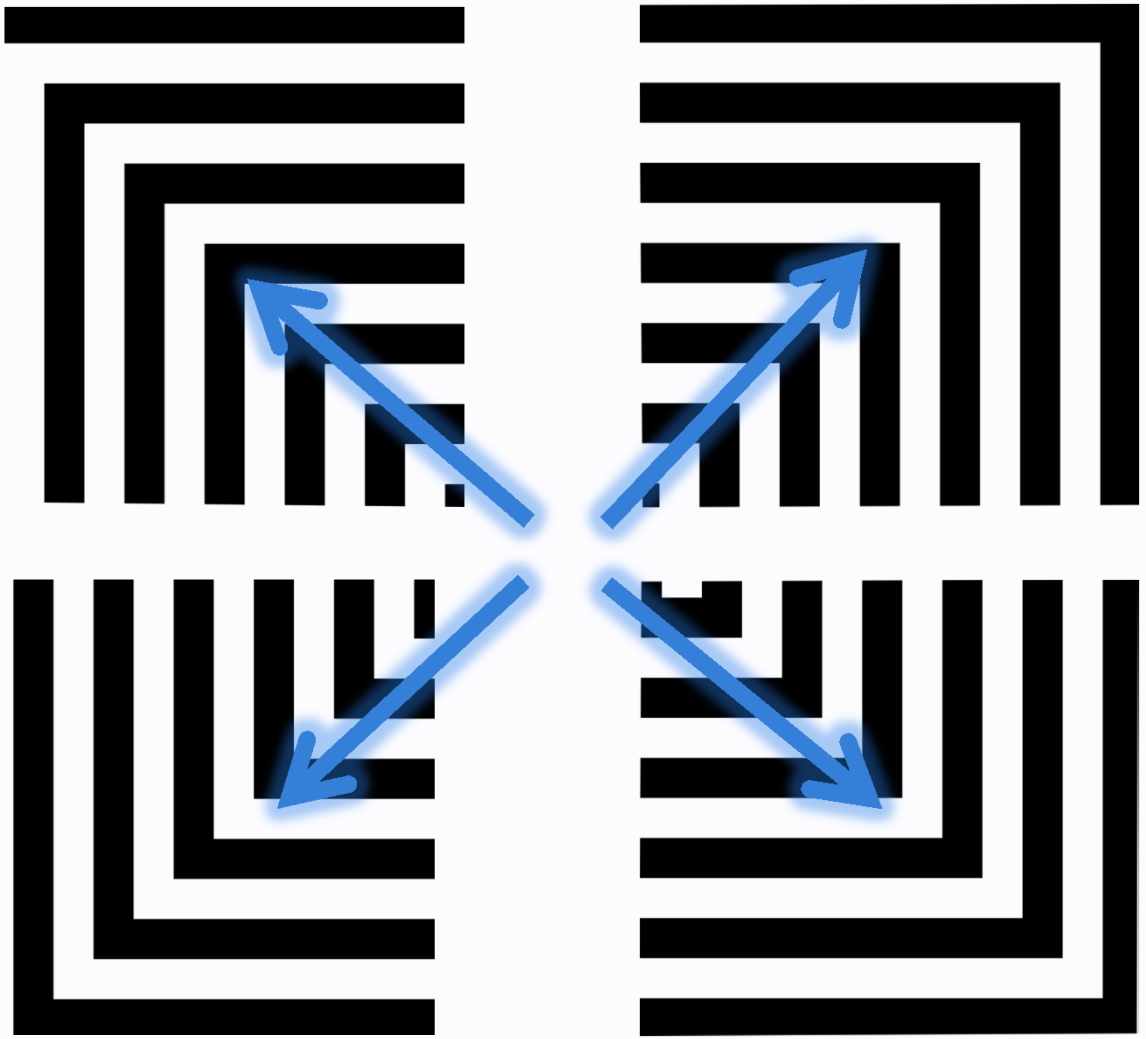
Enhancing the resolution results in adding small-scale modes while having the same large-scale modes



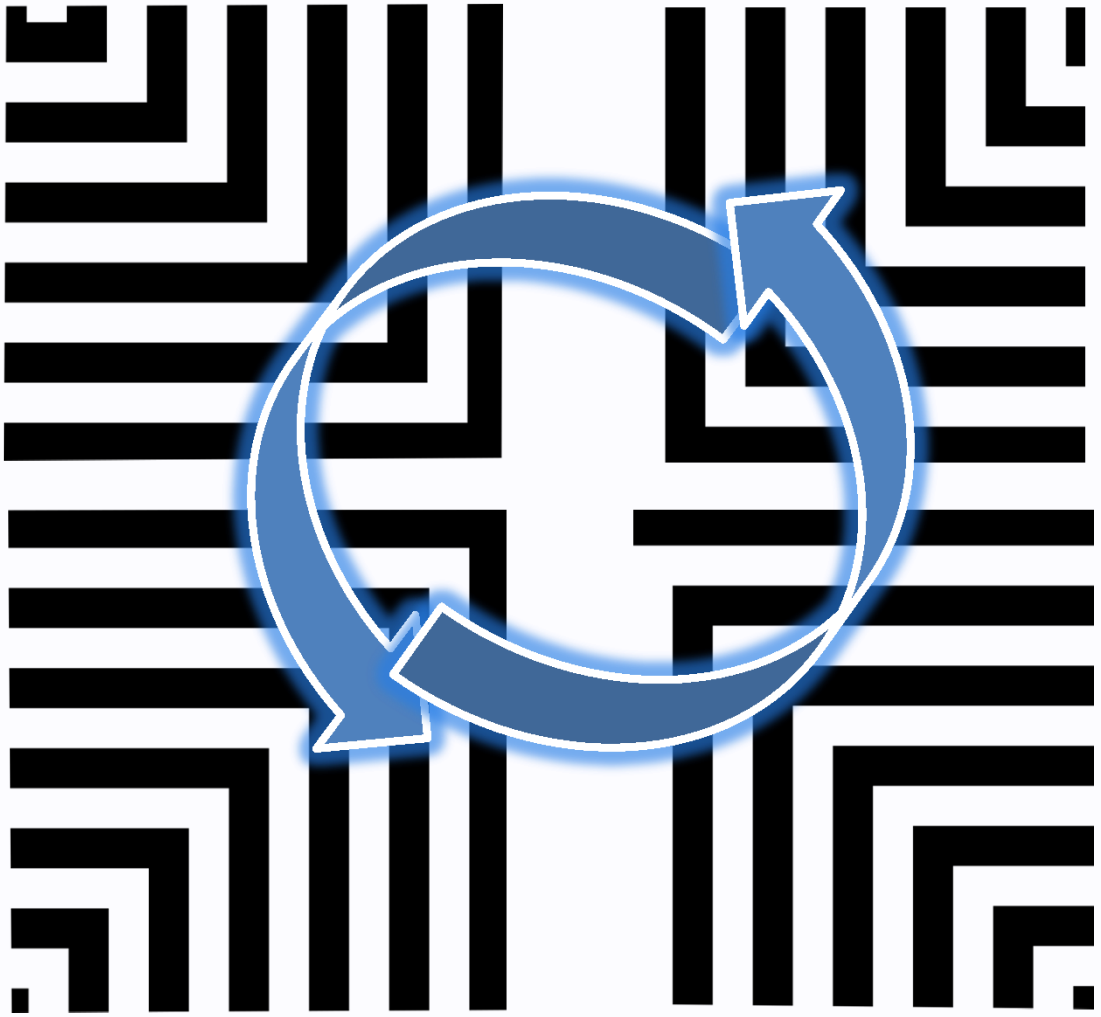
PINOCCHIO



PINOCCHIO

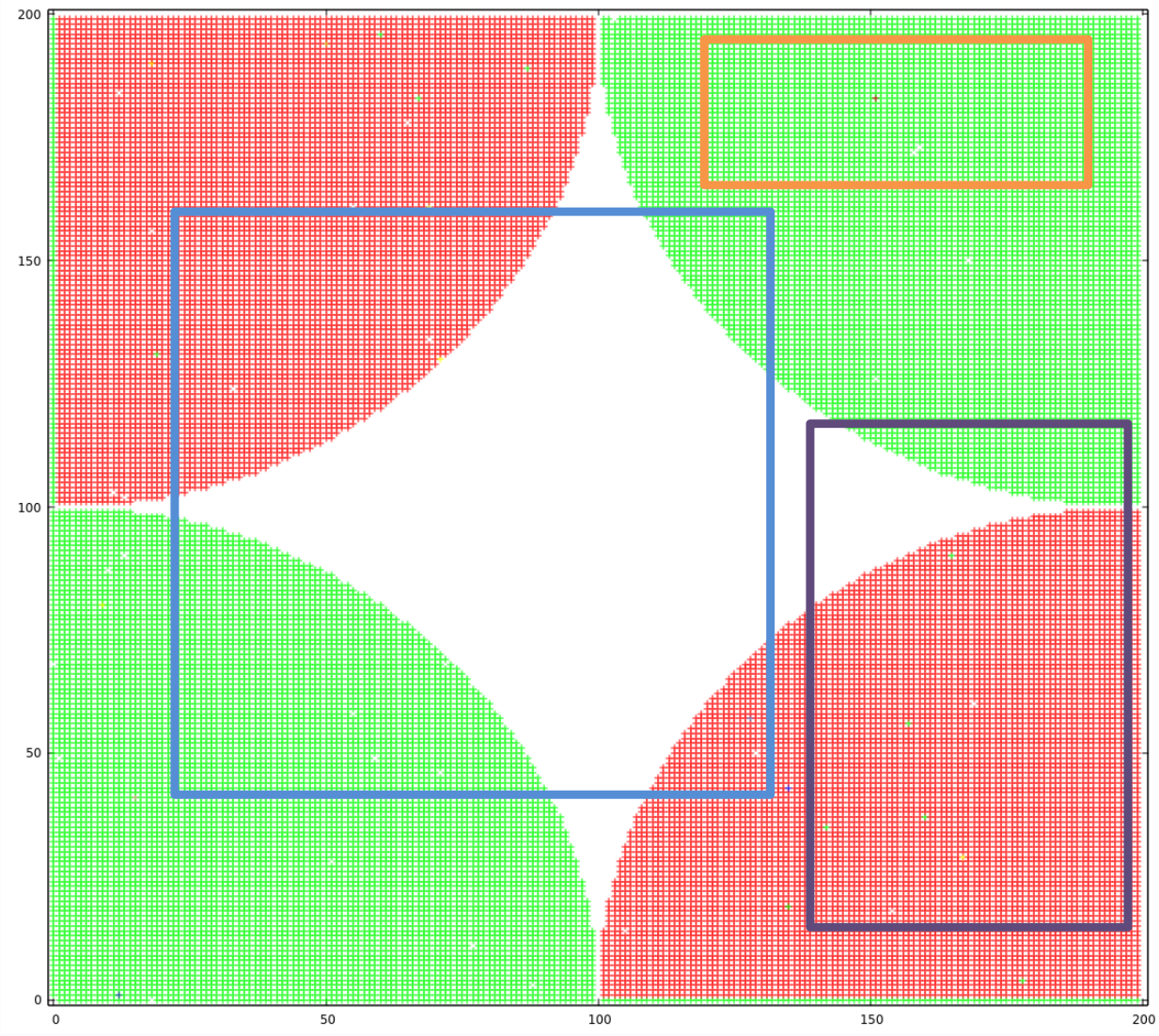


PINOCCHIO

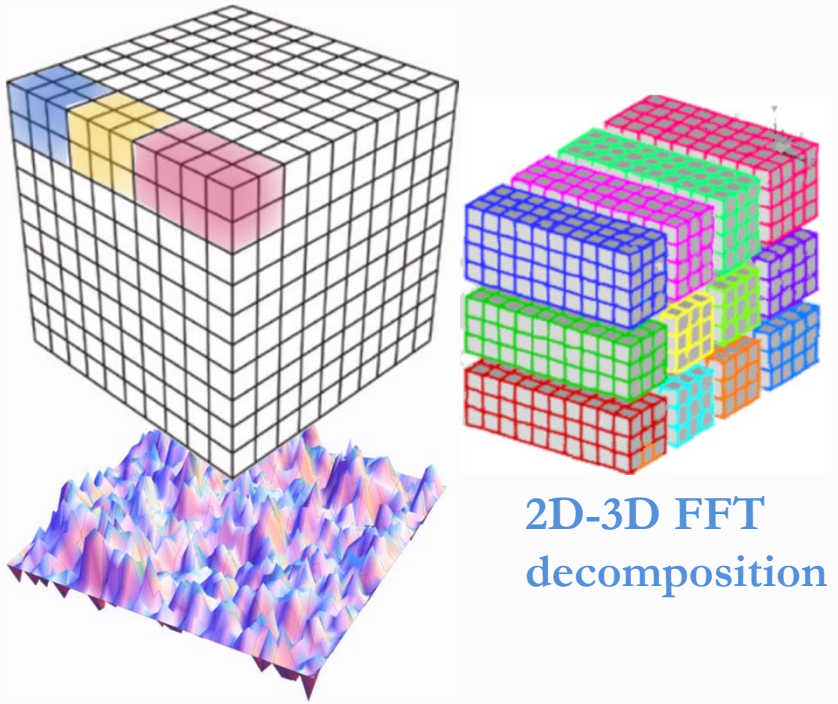


PINOCCHIO

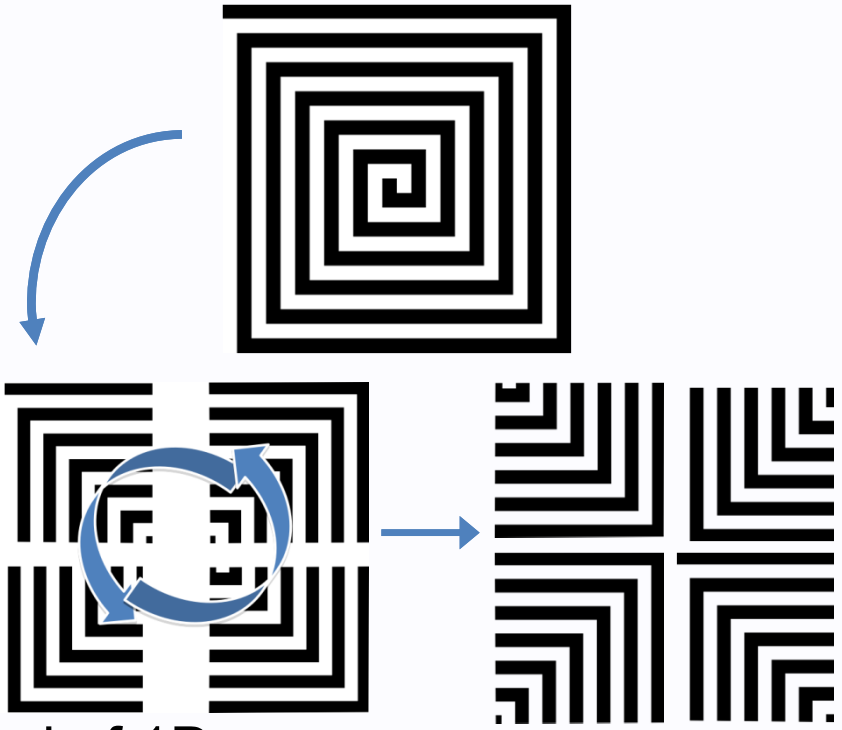
Now we can overcome the memory limits that constrained to 4096^3 @CINECA and generate and manage very large grids



PINOCCHIO



2D-3D FFT
decomposition



- 3D decomposition for FFT, instead of 1D
- completely re-designed algorithm to generate power-spectrum
  - ✓ has same symmetries and properties
  - ✓ each MPI task must have only its portion of the initial random field
- **[ongoing]** *detailed analysis of memory patterns and access*

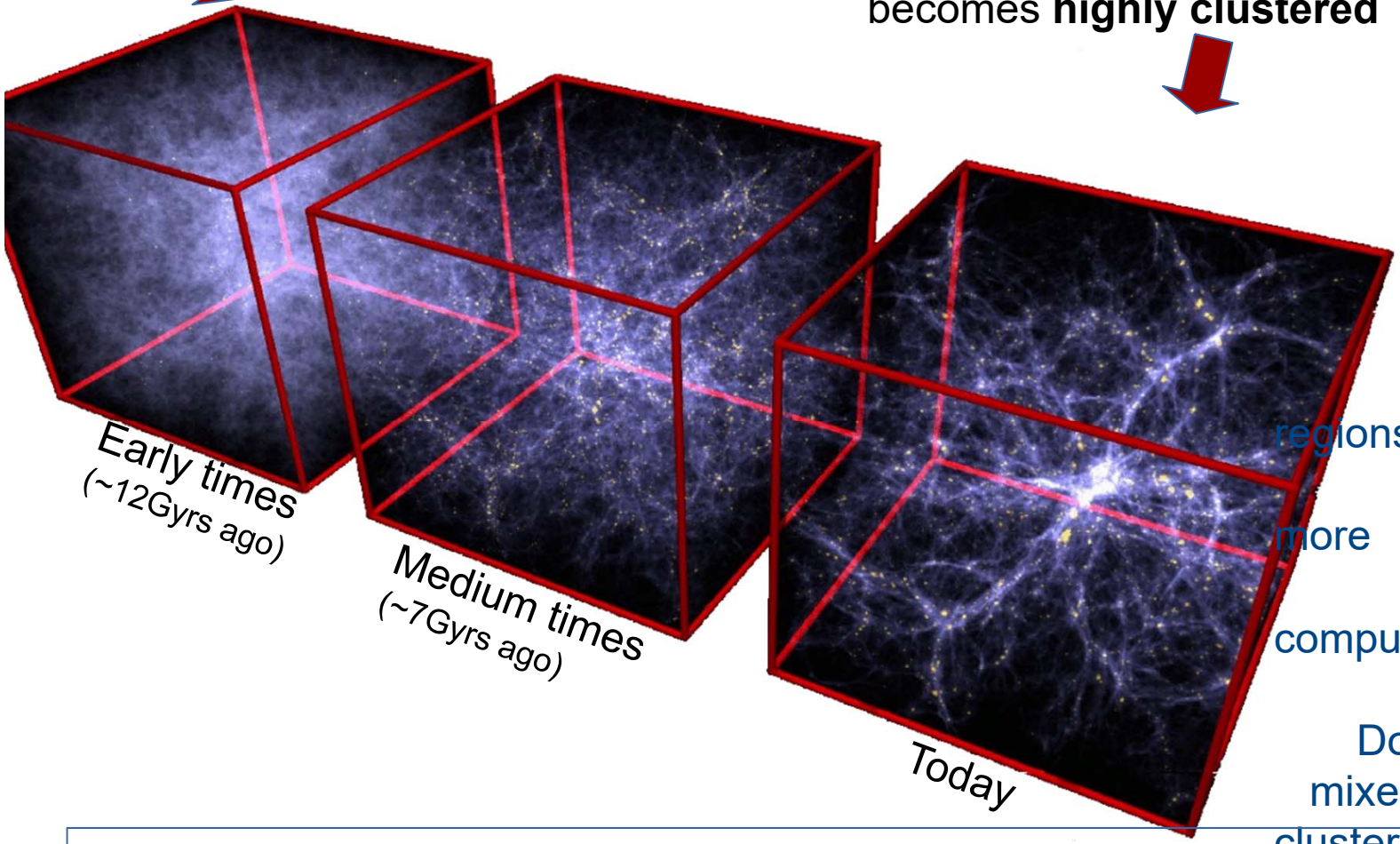
Outline

- Exascale – quick recap
- Project I : PINOCCHIO
- Project II : GADGET
- HPC Ecosystem @ HOTCAT

The initial distribution is **very homegeneous** (P-P force estimates must be very accurate)

There are long-range interactions (i.e. gravity) and short-range ("local") interactions

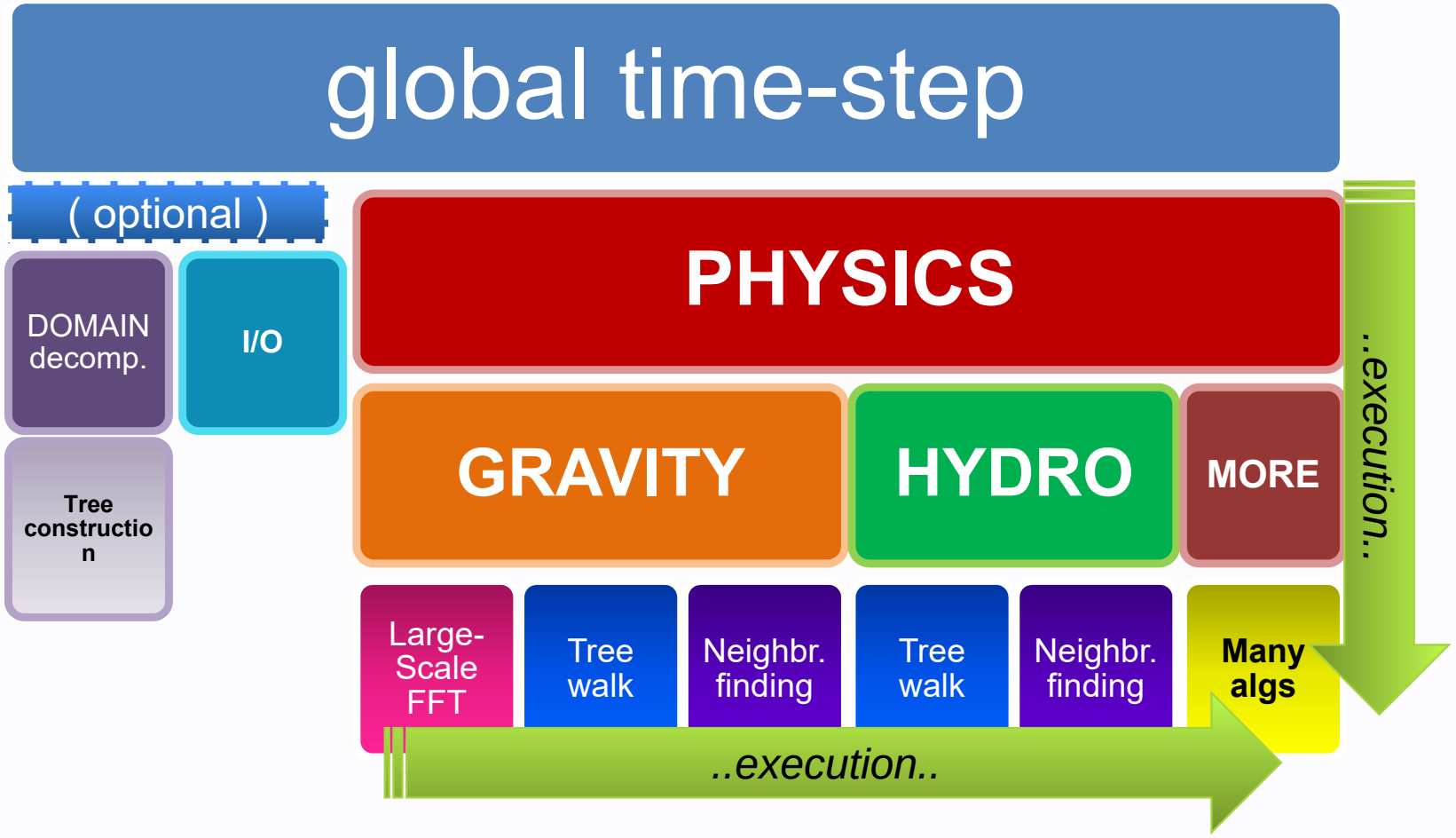
During evolution the particles distribution becomes **highly clustered**



Denser regions are much more computationally expensive. Domain decomp. mixes highly clustered regions and voids. More communication

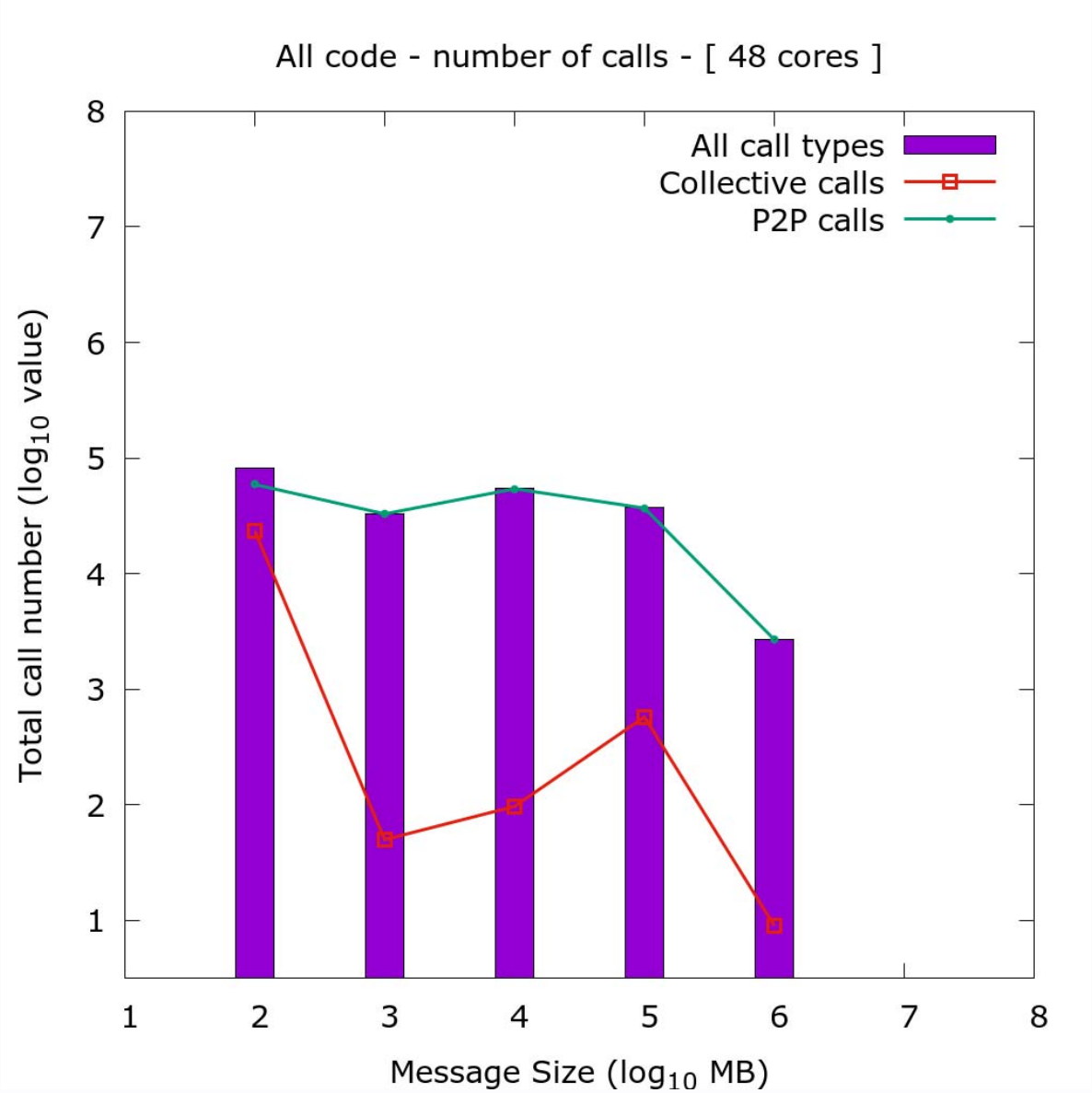
Computational domains are **segments of a Peano curve**, assigned in order to balance both the **work-load** and the **memory-load** (each task pick up alternatively both the most and the less computational intensive segment among the ones in the list)

GADGET

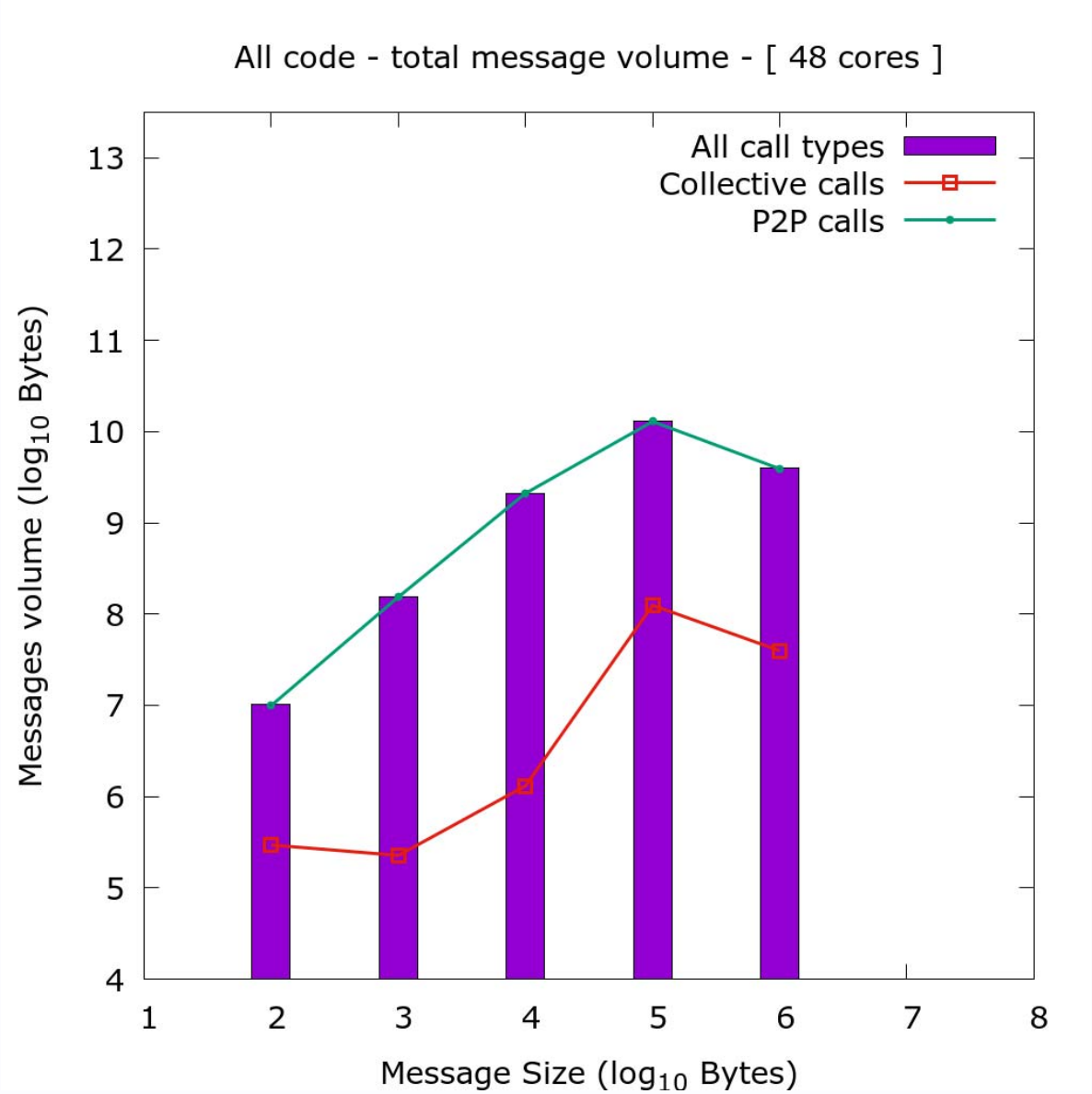


- Extremely complex code, many different algorithms *(long-distance + local physical processes)*
- Rigidly procedural design

GADGET


All code - number of calls - [48 cores]
All call types
Collective calls
P2P calls
Total call number (log10 value)
Message Size (log10 MB)
1
2
3
4
5
6
7
8

GADGET

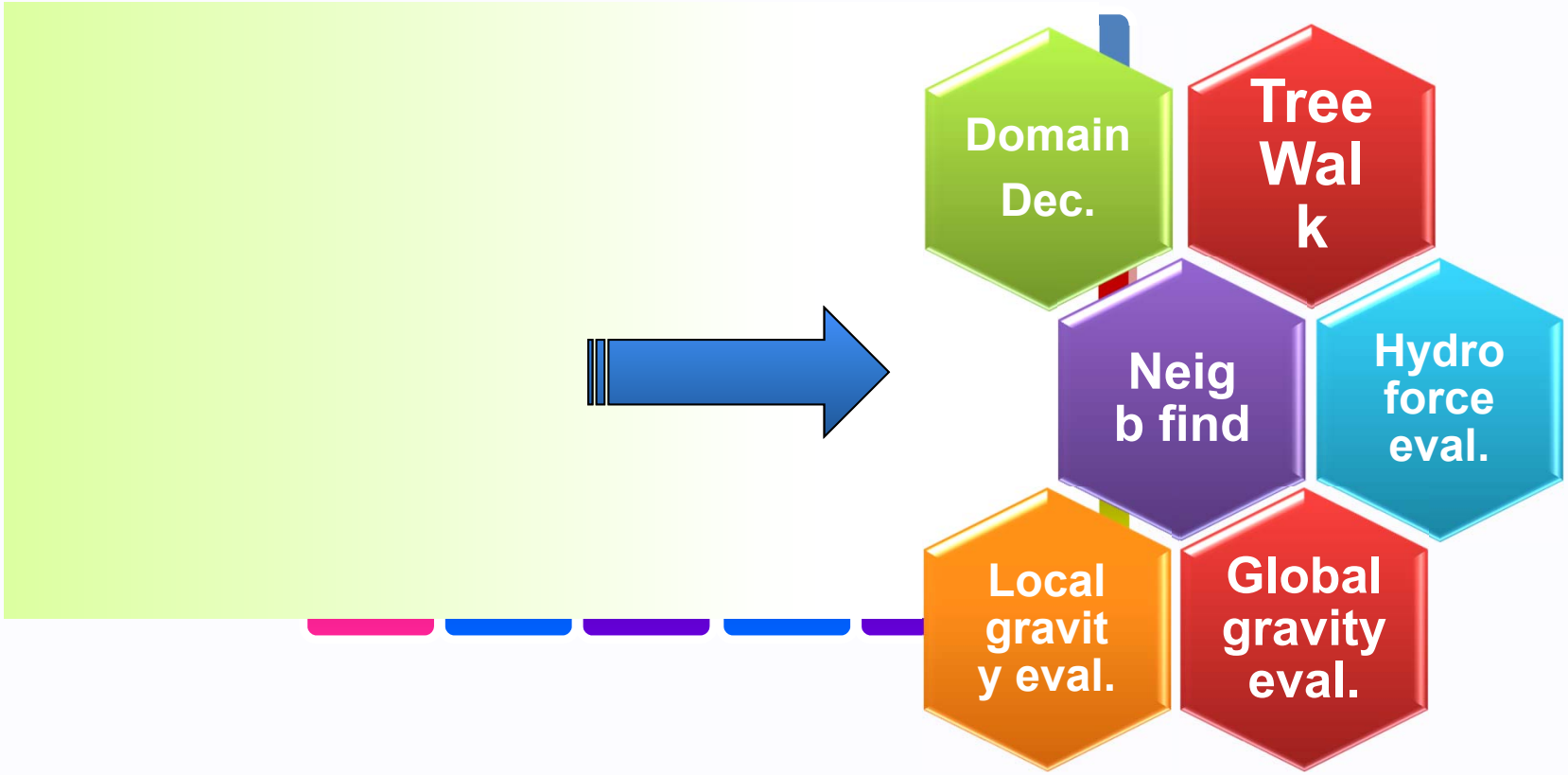

All code - total message volume - [48 cores]
All call types
Collective calls
P2P calls
Messages volume (log10 Bytes)
Message Size (log10 Bytes)
13
12
11
10
9
8
7
6
5
4
1
2
3
4
5
6
7
8

GADGET

GADGET may present some issues

- Relies on a **"monolithic" workflow** instead of being split-up in smallest autonomous "tasks": a relatively low number (~10^3) of MPI threads execute the same work-flow on a fraction of the system
  (OpenMP threads possibly execute concurrently the same task)
- **Adaptively balancing the workload is quite difficult** on million-threads architectures. It might still achieve weak scaling, with increasing parallel inefficiency, but can hardly achieve strong scaling
- It relies on **frequent all-to-all communication / synchronization** cycles
- It is **unaware of heterogeneous memory hierarchy**
- **Communications are mostly "blocking"**
- Data structures are not intrinsically designed to guarantee *(1)* ***cache-efficiency*** and *(2)* ***vectorization-efficiency***

GADGET



- Re-design algorithms in a **task-based, data-driven** perspective
- **Threaded:** as much as possible, **do not** leave any **task idling**
- Native **NUMA-awareness**

Outline

- Exascale – quick recap
- Project I : PINOCCHIO
- Project II : GADGET
- HPC Ecosystem @ HOTCAT

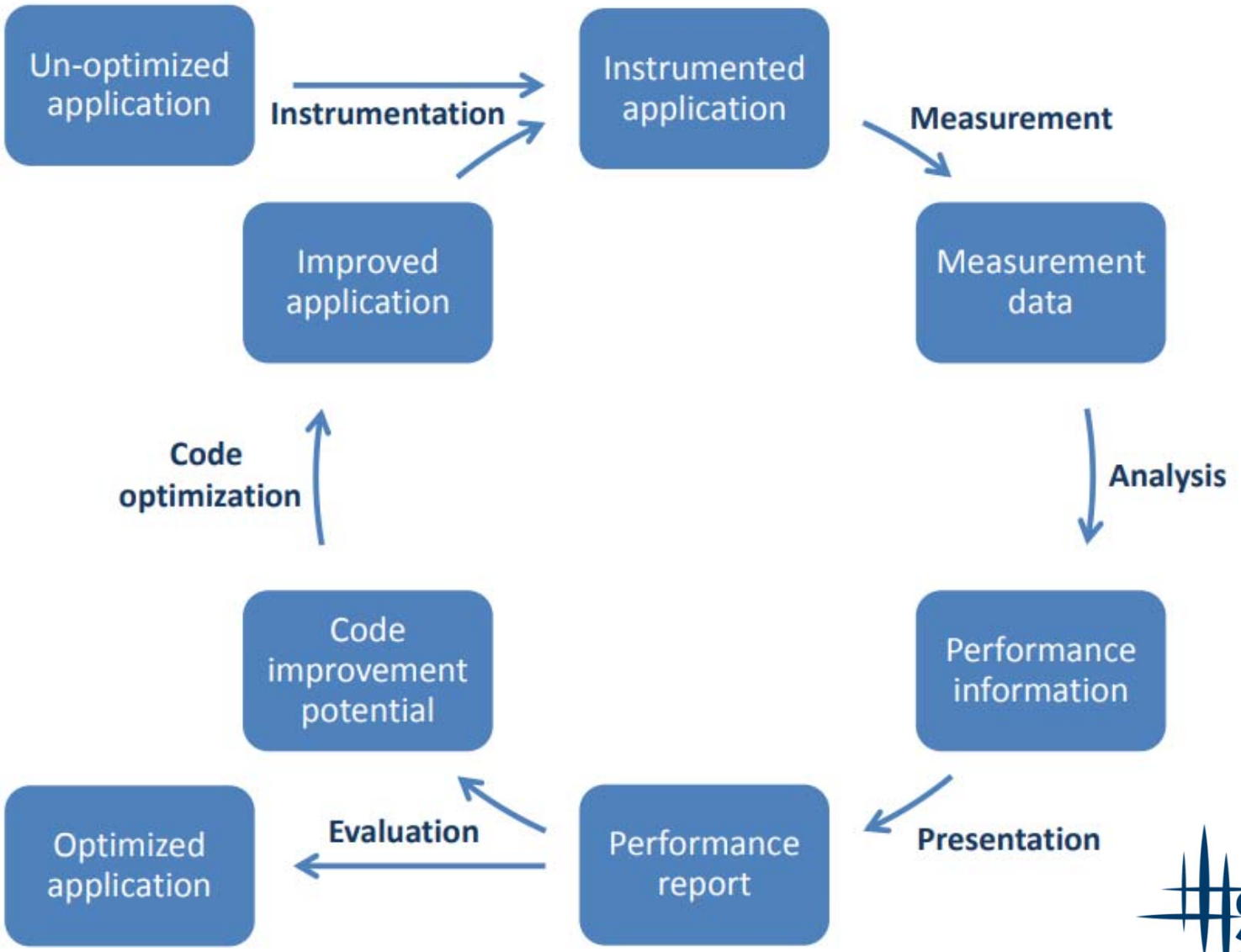
HPC ecosystem @ HOTCAT

Understanding the details of a large parallel/multi-threaded code is not an easy task.

- Bottlenecks
- Inefficiencies
- Deadlocks
- Race conditions
- Errors in memory addressing

- Cache inefficiencies
- Loop optimizations
- Performance counters

HPC ecosystem @ HOTCAT



Score-P
Scalable performance measurement infrastructure for parallel codes

HPC ecosystem @ HOTCAT

There are several very sophisticated open tools, that are mainly outcomes of project research on high-productivity supercomputing.

Many of them are clustered in “ecosystems”, and several of those ecosystems, or some of their components, can talk each other

MpiP

Virtual Inst – High productivity supercomp.

Technische Universität München

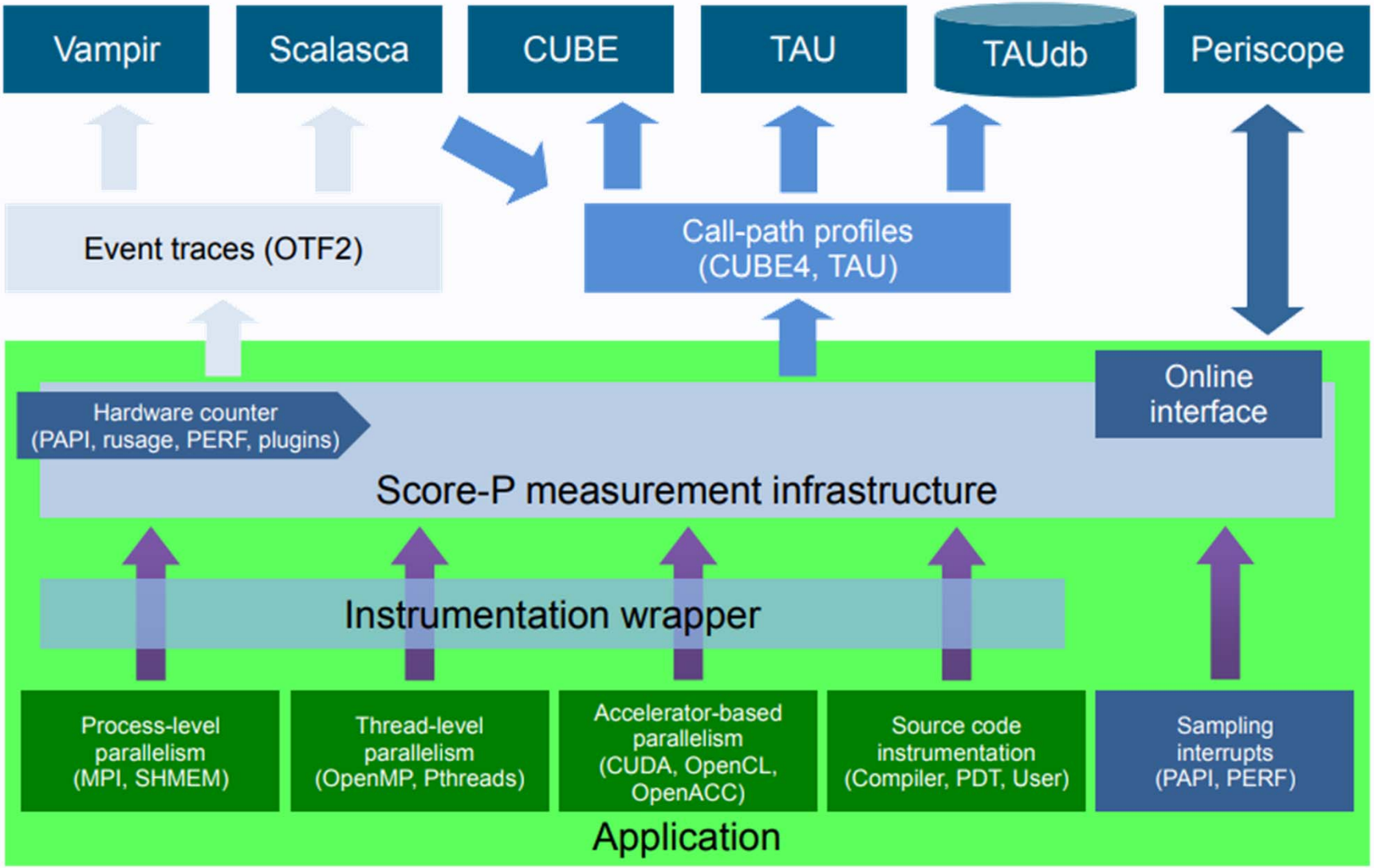
Rice University

Krell Inst., LANL, LLNL, SNL

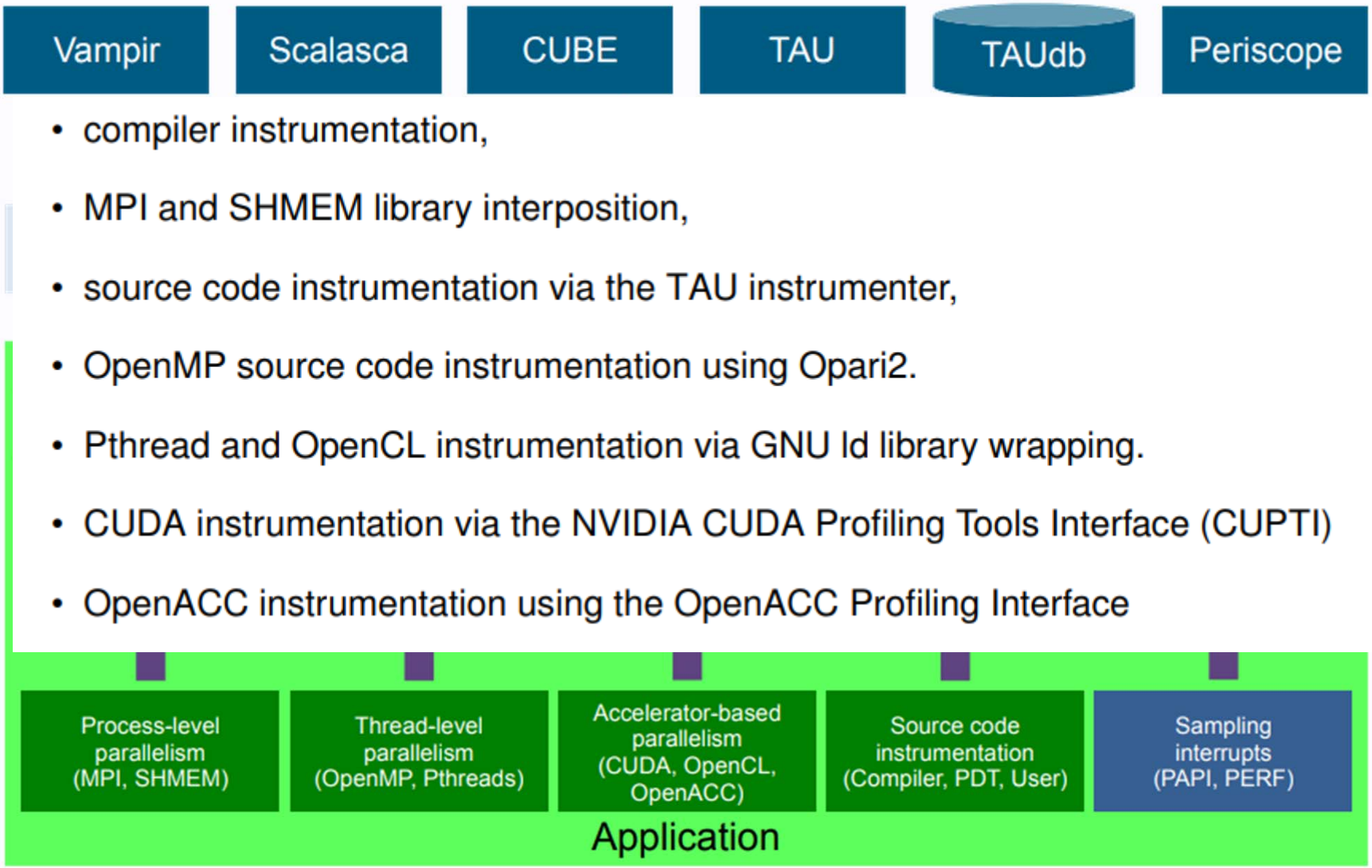
HPC ecocystem @ HOTCAT

```
[ltornatore@bastet ~]$ module av

------------------------------------------------------------ /opt/cluster/Modules/versions -------------------------------------------------------
3.2.10

---------------------------------------------------------- /opt/cluster/Modules/3.2.10/modulefiles -----------------------------------------------
default_gnu   default_intel dot           module-git    module-info   modules       null          use.own

------------------------------------------------------------ /opt/cluster/Modules/3.2.10/compilers -------------------------------------------------
gnu/4.8.5    gnu/4.9.4    gnu/5.4.0    gnu/6.2.0    intel/2017.1

------------------------------------------------------------ /opt/cluster/Modules/3.2.10/libraries -------------------------------------------------
cfitsio/3.410/gnu/4.8.5                 fftw/3.3.5/openmpi/1.8.8/gnu/4.8.5       hdf5/1.8.18/gnu/4.9.4
cfitsio/3.410/gnu/4.9.4                 fftw/3.3.5/openmpi/1.8.8/gnu/4.9.4       hdf5-parallel/1.8.18/openmpi/1.8.8/4.8.5
cfitsio/3.410/gnu/5.4.0                 fftw/3.3.5/openmpi/2.0.1/gnu/4.8.5       hdf5-parallel/1.8.18/openmpi/1.8.8/4.9.4
cfitsio/3.410/gnu/6.2.0                 fftw/3.3.5/openmpi/2.0.1/gnu/4.9.4       hdf5-parallel/1.8.18/openmpi/2.0.1/4.8.5
cfitsio/3.410/intel/2017.1              gsl/2.2/gnu/4.8.5                        hdf5-parallel/1.8.18/openmpi/2.0.1/4.9.4
fftw/2.1.5/openmpi/1.8.8/gnu/4.8.5      gsl/2.2/gnu/4.9.4                        mellanox/hpcx-stack
fftw/2.1.5/openmpi/1.8.8/gnu/4.9.4      gsl/2.2/gnu/5.4.0                        valgrind/3.13.0/openmpi/6.2.0
fftw/2.1.5/openmpi/2.0.1/gnu/4.8.5      gsl/2.2/gnu/6.2.0
fftw/2.1.5/openmpi/2.0.1/gnu/4.9.4      hdf5/1.8.18/gnu/4.8.5

---------------------------------------------------------------- /opt/cluster/Modules/3.2.10/mpi ---------------------------------------------------
intel/impi/2017.1        mvapich/2.2/gnu/4.9.4    openmpi/1.8.8/gnu/4.9.4  openmpi/2.0.1/gnu/4.8.5  openmpi/2.1.2/gnu/6.2.0
mellanox/hpcx-ompi-v1.10 mvapich/2.2/gnu/5.4.0    openmpi/1.8.8/gnu/5.4.0  openmpi/2.0.1/gnu/4.9.4  openmpi/3.0.0/gnu/6.2.0
mvapich/2.2/gnu/4.8.5    openmpi/1.8.8/gnu/4.8.5  openmpi/1.8.8/gnu/6.2.0  openmpi/2.0.1/gnu/6.2.0

------------------------------------------------------------ /opt/cluster/Modules/3.2.10/python ----------------------------------------------------
exochipp      exochipp5.4.0 python3.6.1

------------------------------------------------------------ /opt/cluster/Modules/3.2.10/tools -----------------------------------------------------
cube-4.3.5/gnu/6.2.0          libmonitor/gnu/6.2.0          likwid-4.2.0/gnu/6.2.0        otf2-2.1/gnu/6.2.0            scalasca-2.3.1/gnu/6.2.0
gperftools/gnu/6.2.0          libpfm-4.8.0/gnu/6.2.0        opari2-2.0.2/gnu/6.2.0        papi-5.5.1/gnu/6.2.0          scorep-3.1/gnu/6.2.0
hpctoolkit/gnu/6.2.0          libunwind-0.99-beta/gnu/6.2.0 oprofile-1.2.0/gnu/6.2.0      papiex-1.0.3/gnu/6.2.0        xerces-3.2.0/gnu/6.2.0
[ltornatore@bastet ~]$
```

HPC ecosystem @ HOTCAT



HPC ecosystem @ HOTCAT



HPC ecosystem @ HOTCAT

Score-P offers you a quite flexible measurement infrastructure