GRCHOMBO - Using the Cluster

How to use?

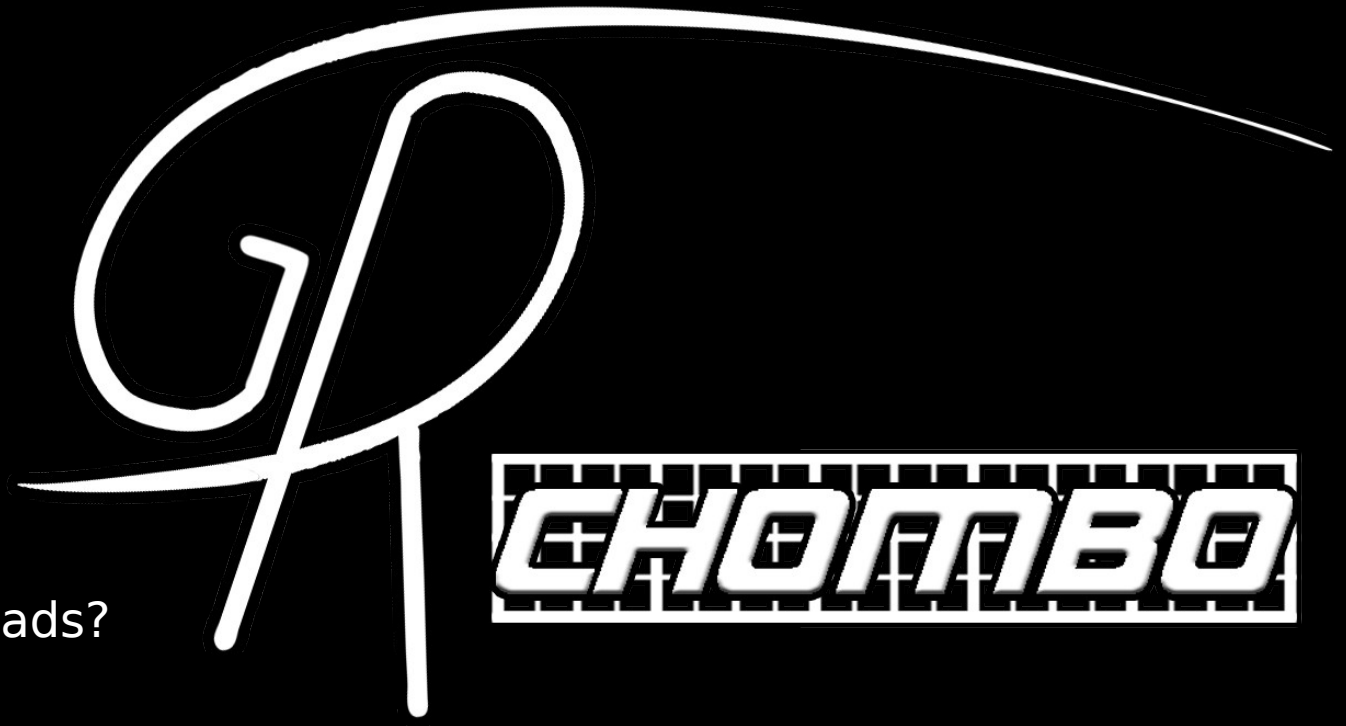
1) What are nodes and cores?

2) OpenMP vs MPI

3) What are ranks and threads?

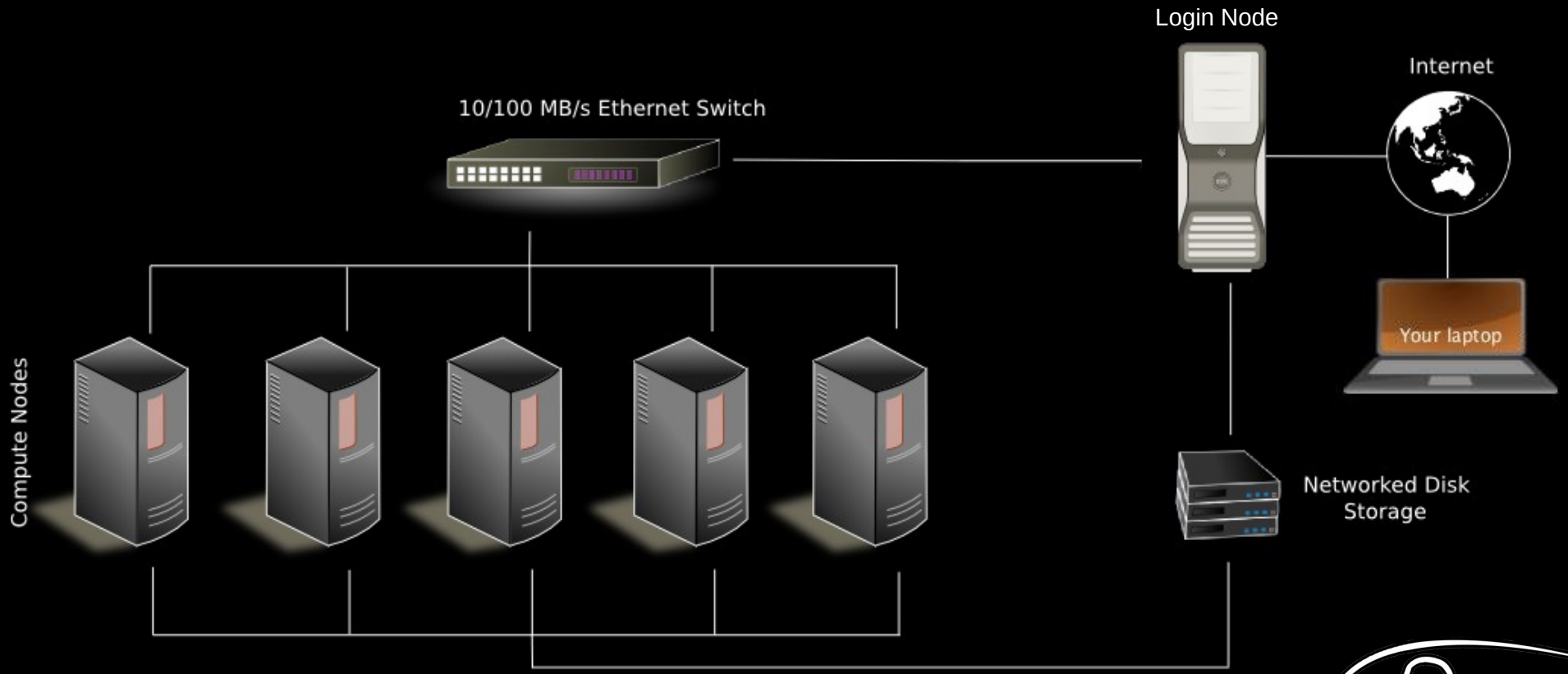
4) How to decide #nodes/cores/ranks/threads?

5) Load balancing



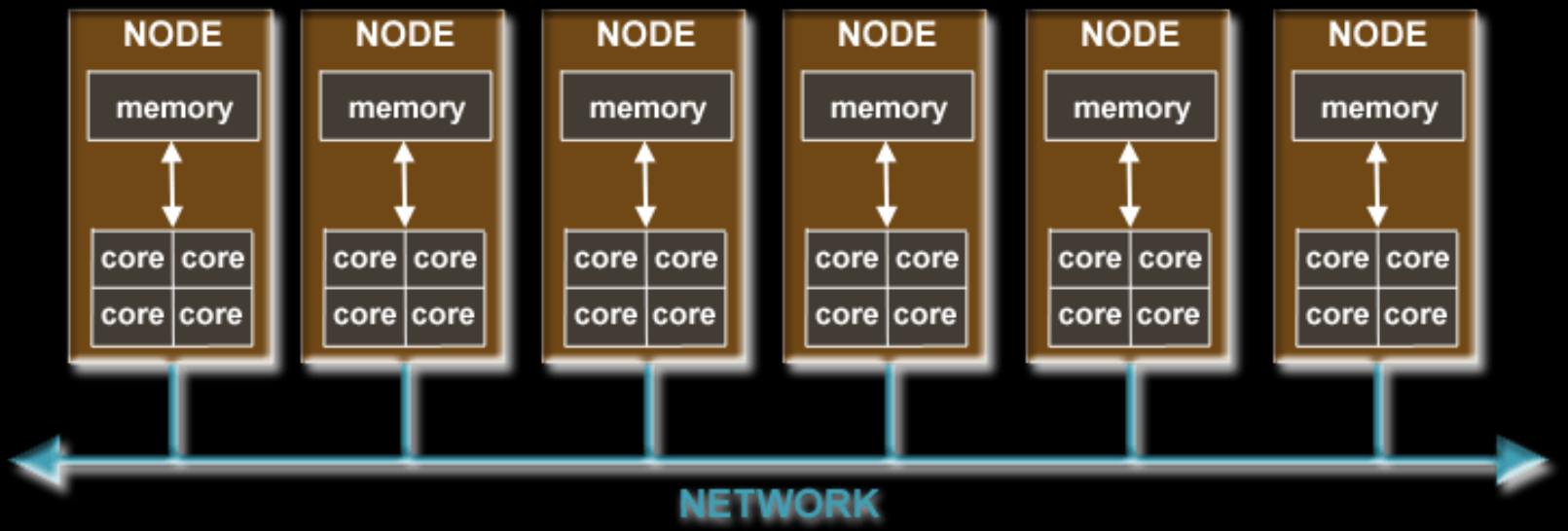
Tiago França, School of Mathematical Sciences, Queen Mary University of London

The Cluster



Original image by Princeton University

Nodes vs Cores



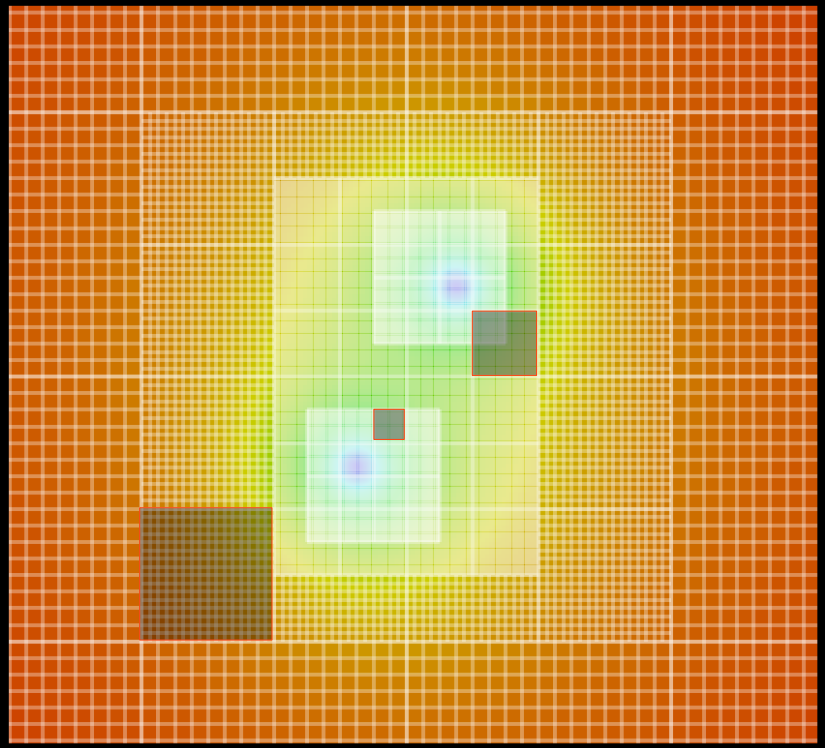
- Nodes have to communicate through the network
- Cores wihin he same node:
  1) can communicate directly (==> faster)
  2) have shared memory (==> faster)

Original image by Princeton University

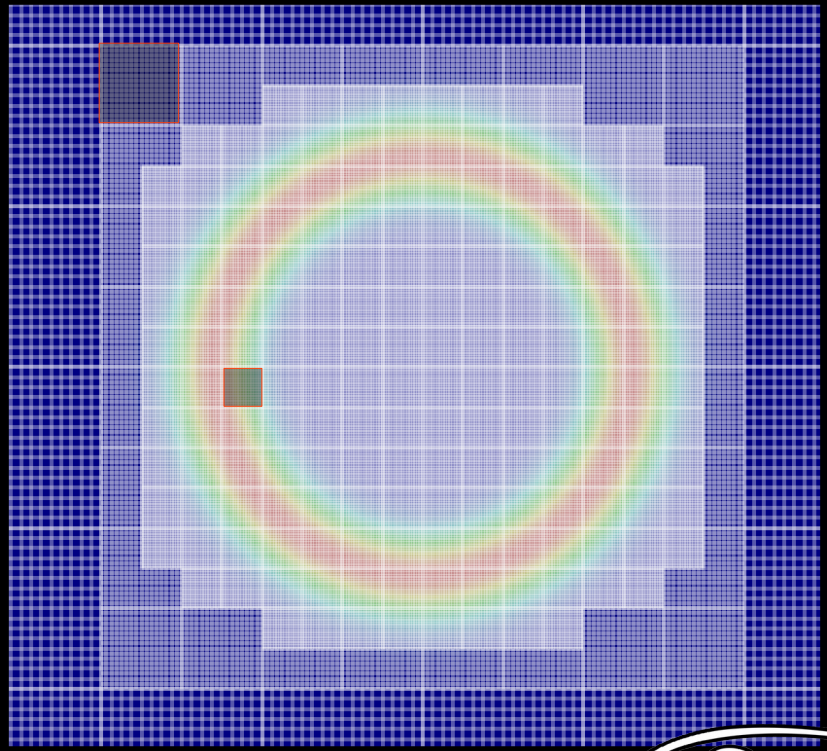
CHOMBO

GRChombo shares boxes across CPUs

Binary Black Hole Example



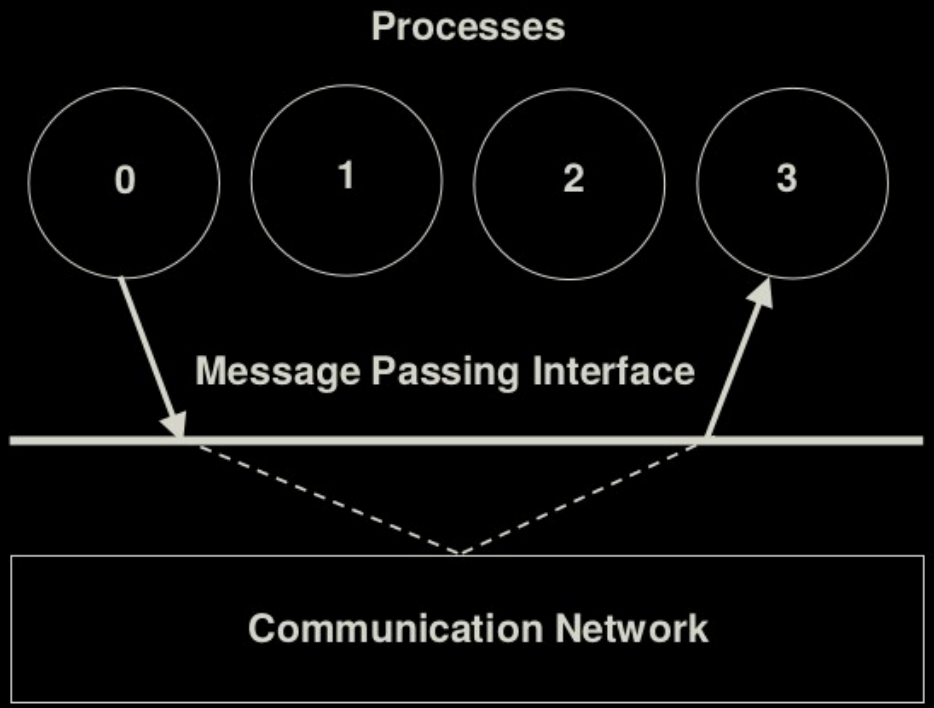
Scalar Field Example



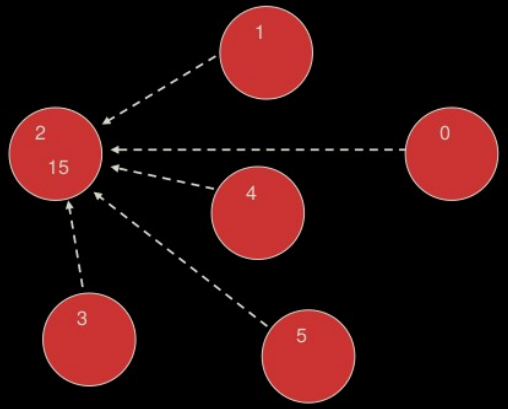
All boxes are spread evenly across parallel processes.

MPI 101

MPI (Message Passing Interface) is an interface for communication between processes (ranks!) across distributed memory



Example of ‘Reduction’ - global sum

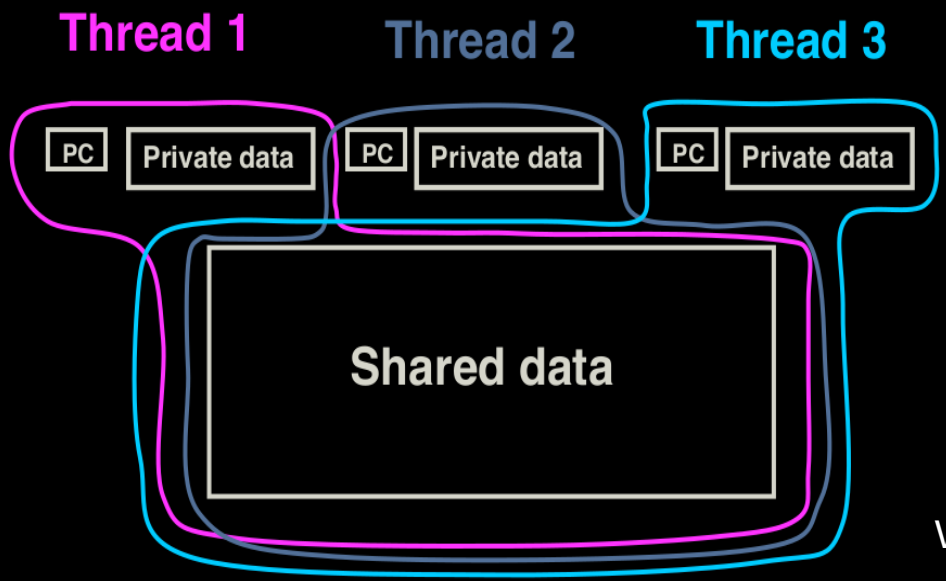


Where can you find it in the code?

1) In Chombo:
   - Useful functions in BaseTools → SPMD.H
   - Other files as BoxTools → BoxLayoutDataI.H
   - (it’s a dangerous road to go there!)
2) In GRChombo
   - GRChomboCore → SetupFunctions.hpp
   - AMRInterpolator → MPIContext.hpp
   - (and others...)

OpenMP 101

OpenMP (Open Multi-Processing) is an interface for parallel programming (threads!) with shared memory



Example of FOR loop

```for (int i = 0; i < n; ++i) {
  // do stuff
}```

C++

```#pragma omp parallel for
for (int i = 0; i < n; ++i) {
  // do stuff in parallel
}```

OpenMP & C++

Where can you find it in the code?
BoxLoops → BoxLoops.impl.hpp
(mostly)

Sample Job Script

```
#!/bin/bash
#SBATCH -p (partition)
#SBATCH -A (account)
#SBATCH --time=12:00:00
#SBATCH --job-name=sample

#SBATCH --nodes=10
#SBATCH --ntasks-per-node=8
#SBATCH --cpus-per-task=4

export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK

# other stuff? Import modules?
# (...)

mpiexec program params.txt
```

(this is for a slurm job script, even though similar in other managers as pbs)

#ranks/node * #threads/rank = #cpus/node
(32 in this case)

#ranks/node * #nodes = #ranks
(80 in this case)

nodes
MPI ranks per node
OpenMP threads per MPI rank

(tell OpenMP about it)

To run with MPI, you might see
“mpiexec / mpirun / srun / … ”

Load Balancing

You run a simulation. What to look for?

pout.0

```
GRAMRLevel::advance level 0 at time 8.75 (2.13795 M/hr). Boxes on this rank: 1.
GRAMRLevel::advance level 1 at time 8.75 (2.13783 M/hr). Boxes on this rank: 1.
GRAMRLevel::advance level 2 at time 8.75 (2.13772 M/hr). Boxes on this rank: 1.
GRAMRLevel::advance level 3 at time 8.75 (2.13754 M/hr). Boxes on this rank: 1.
GRAMRLevel::advance level 4 at time 8.75 (2.13743 M/hr). Boxes on this rank: 1.
GRAMRLevel::advance level 5 at time 8.75 (2.13724 M/hr). Boxes on this rank: 1.
GRAMRLevel::advance level 6 at time 8.75 (2.13705 M/hr). Boxes on this rank: 1.
GRAMRLevel::advance level 6 at time 8.75391 (2.13781 M/hr). Boxes on this rank: 1.
GRAMRLevel::advance level 5 at time 8.75781 (2.13856 M/hr). Boxes on this rank: 1.
GRAMRLevel::advance level 6 at time 8.75781 (2.13838 M/hr). Boxes on this rank: 1.
```

the level #boxes

pout.239

```
GRAMRLevel::advance level 0 at time 8.75 (2.13765 M/hr). Boxes on this rank: 0.
GRAMRLevel::advance level 1 at time 8.75 (2.13765 M/hr). Boxes on this rank: 0.
GRAMRLevel::advance level 2 at time 8.75 (2.13765 M/hr). Boxes on this rank: 0.
GRAMRLevel::advance level 3 at time 8.75 (2.13765 M/hr). Boxes on this rank: 0.
GRAMRLevel::advance level 4 at time 8.75 (2.13765 M/hr). Boxes on this rank: 0.
GRAMRLevel::advance level 5 at time 8.75 (2.13765 M/hr). Boxes on this rank: 0.
GRAMRLevel::advance level 6 at time 8.75 (2.13765 M/hr). Boxes on this rank: 0.
GRAMRLevel::advance level 6 at time 8.75391 (2.13861 M/hr). Boxes on this rank: 0.
GRAMRLevel::advance level 5 at time 8.75781 (2.13956 M/hr). Boxes on this rank: 0.
GRAMRLevel::advance level 6 at time 8.75781 (2.13956 M/hr). Boxes on this rank: 0.
```

Goals:

1) not too many boxes per rank
2) as few ranks with no boxes as possible

Note: what matter most is the finest level (6 in this case), as it's the one that runs more often. The levels above can have no boxes in some ranks.

Look at pout.0 → too many boxes?

Look at last pout's → are they empty? How many are empty?

(LB.txt might have some relevant info if you explore)

Load Balancing

Too many boxes? Solutions (“This is not an exact science”, Miren Radia):

- Option 1: increase #nodes
- Option 2: reduce #threads/rank (openMP) and increase #ranks/node (MPI)
- Option 3: decrease resolution
- Option 4: increase box size

Note 1: OpenMP is slower than MPI in GRChombo (beyond 4 OpenMP threads, scaling is not good), so pick option 2 instead of 1 only if you want to save CPU hours, if queue waiting time is too big or simply can’t ask for more nodes.
Note 2: sometimes you might experience error messages saying you ran out of memory. That is equivalent to having too many boxes, so proceed as above.

Too few boxes?

Do the opposite of the options above, but typically option 1 (saves cpu hours!!) or 2 will be enough for this case.

Option 3: Change Resolution

Things you can do:

A) Change #levels (max_level parameter)

B) Change N1-N3 (make the grid coarser/finer)

C) Change N1-N3 and L accordingly (make grid smaller/bigger)

D) Change regridding threshold

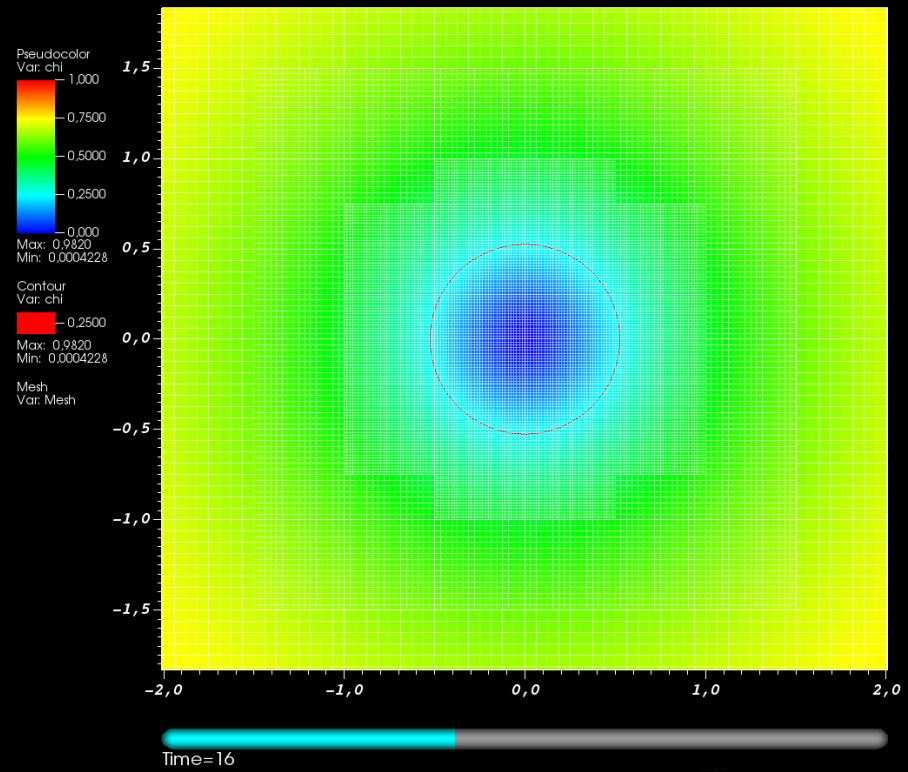
E) [advanced] Change tagging criteria

A few extra tips:

- Can you apply some symmetric boundary conditions to your problem?

- If you are doing black hole simulations, ensure ~40 points across the horizon. If you have more/less, then think of options A/B/C above.

Note: an easy way to find the horizon is to plot a contour of ‘chi’ (after puncture gauge settles, that is at about 0.25 for 0 spin, and closer to 0.15 for a high 0.8 spin)



Option 4: Change Box Sizes

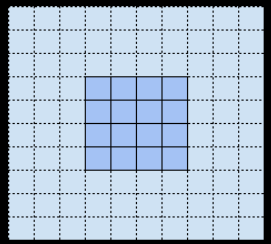
params.txt

```
max_level = 9
regrid_interval = 0 0 0 64 64 64 64 64 64 0

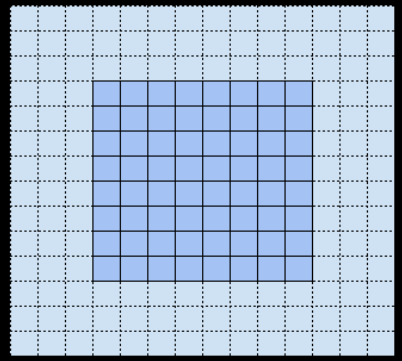
# Max and min box sizes
max_grid_size = 16
block_factor = 16
```

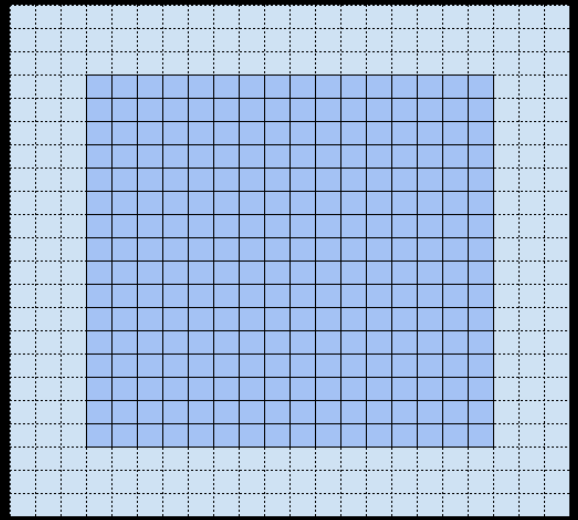
Box Size	% without ghosts
4	6%
8	19%
16	38%
32	60%

4x4(x4)



8x8(x8)





16x16(x16)

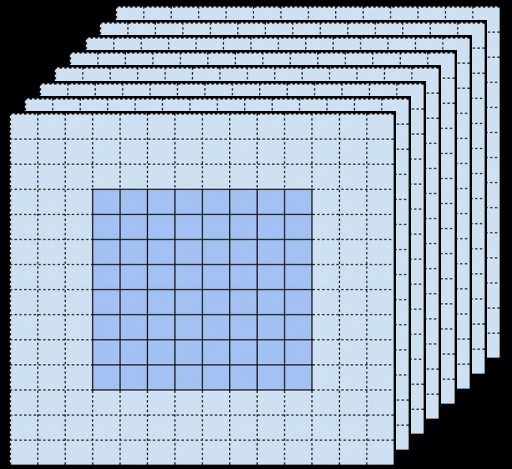
CHOMBO

Option 4: Change Box Sizes

+ Box Size ==> - communication
- #ghosts
but + memory/rank (if you have few boxes, otherwise not true)
& - #boxes ==> can't use as many MPI ranks

Recommended: 8, 16, 24 or 32
(multiples of 8, to match vectorization)

If you have very few boxes, switch to 8
If you have too many boxes, switch to 16/32



CHOMBO

Key Points

- Don’t forget to set OMP_NUM_THREADS

- Try to choose a reasonable #nodes, #ranks, #threads
  ||
  v
- Look at your pout files (or LB.txt). Too many boxes? Too few boxes?
  ||
  v
- Balance MPI / openMP / resolution / box size