GRChombo job scripts OpenMP/MPI

Load balancing

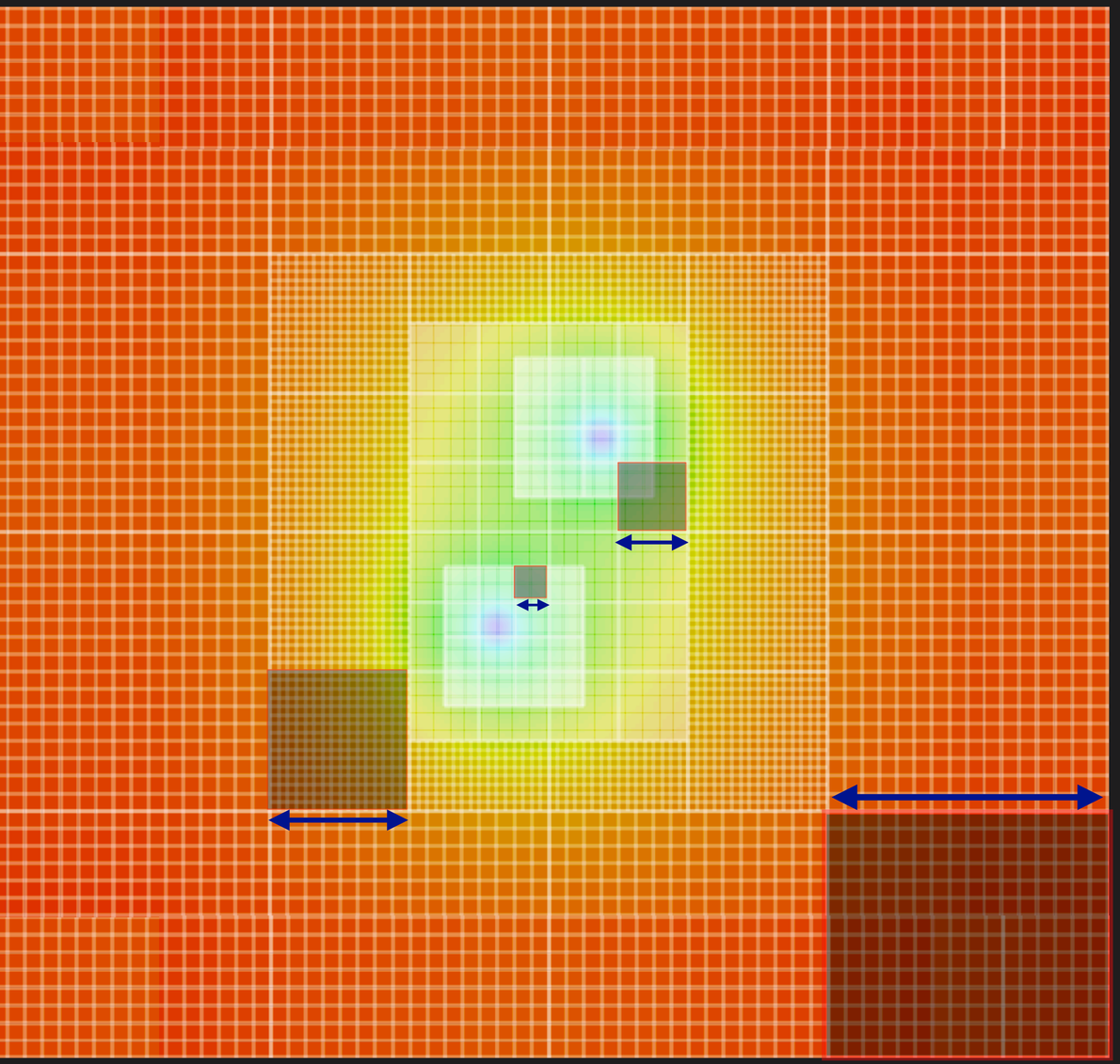
Dina Traykova

GRChombo meeting, 1st Apr 2022

(Adapted from Tiago França, some previous GRChombo meeting)

Some relevant grid parameters

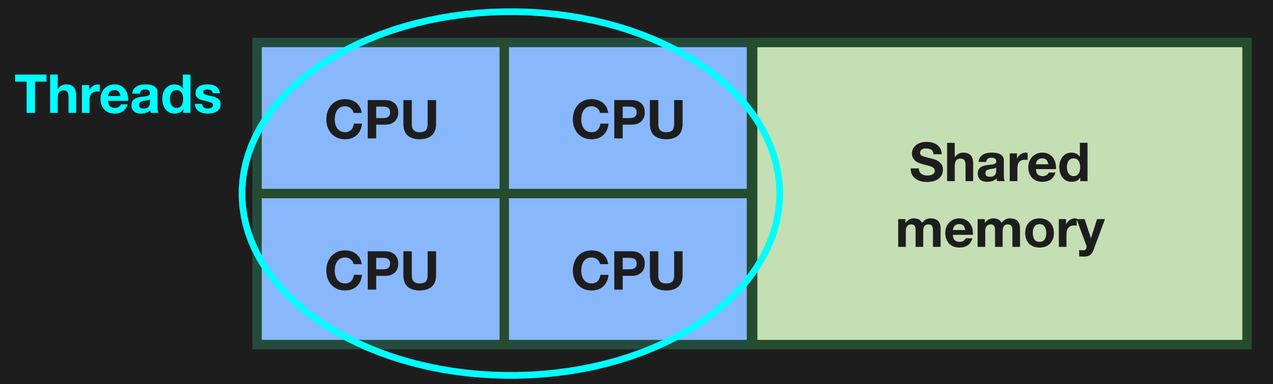
Binary BH example



- Number of cells along each dimension, (N1, N2, N3)
- Grid length, L
- Grid spacing on the coarsest level, $\Delta x = L/N_{max}$
- max_level (fixed refinment = 2:1)
- max_box_size, min_box_size
  - GRChombo shares boxes across CPUs
  - N should be be a multiple of
- #boxes_coarsest = (N/max_box_size)^3

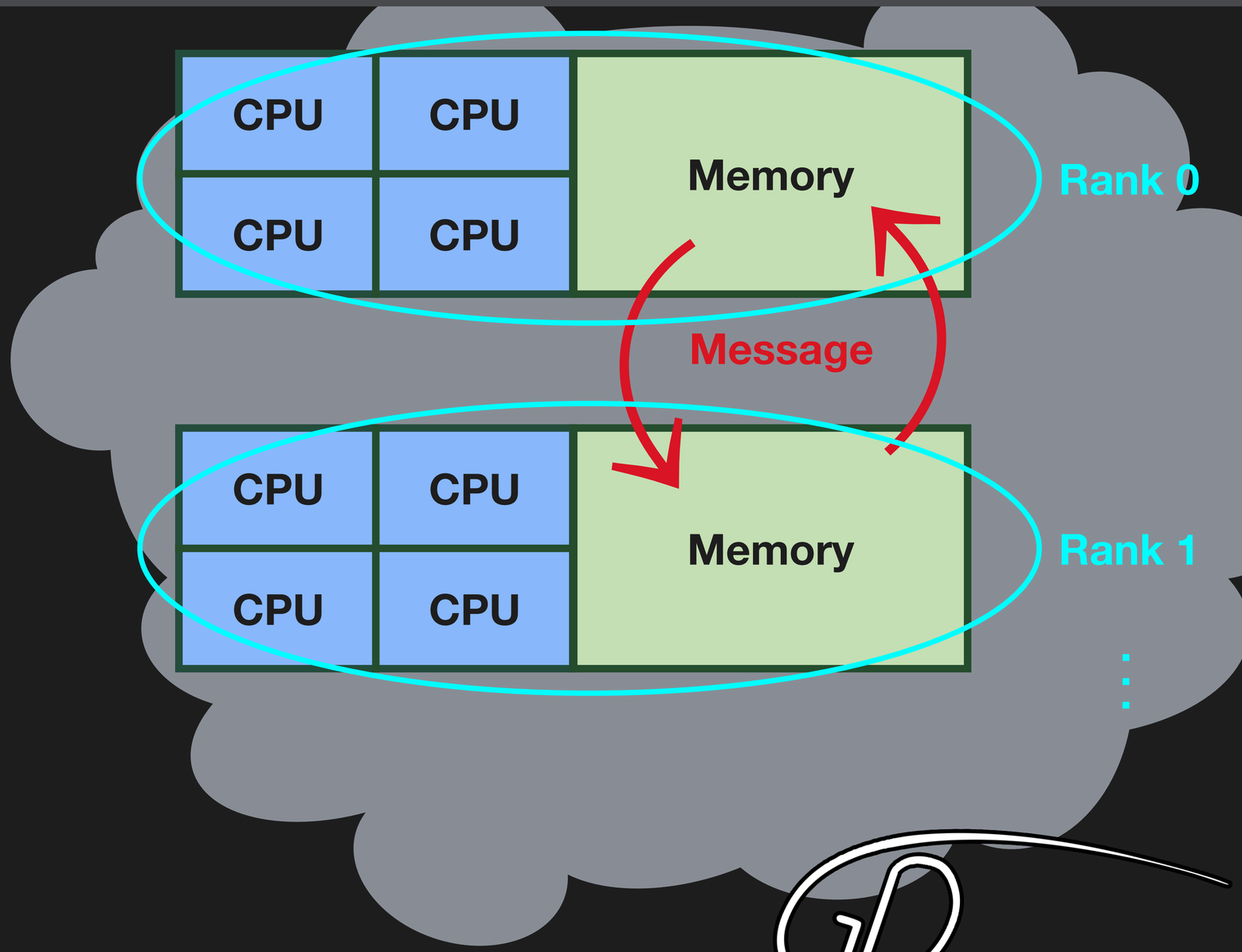
OpenMP (Open multi-processing)

- Designed for shared memory
- Single system with multiple cores (threads) sharing memory
- Process: an executing instance of program
- Thread: a subset of a process, shares resources with other threads and the parent process
- OpenMP mostly in Source/BoxUtils/BoxLoops.impl.hpp



MPI (Message Passing Interface)

- Designed for distributed memory
- Multiple systems
- Send/receive messages
- Slower communication than with OpenMP
- In GRChombo – better parallelisation than OpenMP
- In general – better to allocate more processes than threads



Example job script

- GRChombo executables can be run in parallel:
  `mpirun -np 4 ./Main_BinaryBH3d.Linux.64.mpiicpc.ifort.OPTHIGH.MPI.OPENMPCC.ex params.txt`
- But on cluster – job script:

```
#!/bin/bash -l
# Standard output and error:
#SBATCH -o ./out.%j
#SBATCH -e ./err.%j
# Initial working directory:
#SBATCH -D ./
# Job Name:
#SBATCH -J BBH
#
# Number of nodes and MPI tasks per node:
#SBATCH --nodes=10
#SBATCH --ntasks-per-node=10
# for OpenMP:
#SBATCH --cpus-per-task=4
#
#SBATCH --mail-type=none
#SBATCH --mail-user=
#
# Wall clock limit:
#SBATCH --time=24:00:00

# Load compiler and MPI modules with explicit version specifications,
# consistently with the versions used to build the executable.
module purge
module load ...

# Export any libraries necessary
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/...

export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK

# Run the program:
srun /path/to/program.ex params.txt
```

#nodes - here 10

ntasks-per-node = # MPI ranks per node

cpus-per-rank = # OpenMP threads per MPI rank

#ranks/node * #nodes = #ranks (here 100)

Note: Should always have:

#ranks/node * #threads/rank = #cpus/node (here 40)

! Remember to set OMP_NUM_THREADS

Load balancing

- All processes should have ~ same computational load
- On the coarsest grid: #boxes = (N/max_box_size)^3
- 1 – a few boxes per process
- On the finer levels #boxes – not easy to predict ahead of time
- Calculation much more complicated – but will be more
- Some trial and error: run a few steps -> check how well the sims are balanced.. adjust nodes/threads/ rank as needed

Miren Radia: “This is not an exact science”

CHOMBO

Load balancing

```
GRAMRLevel::advance level 0 at time 2 (63.8269 M/hr). Boxes on this rank: 1 / 32
GRAMRLevel::advance level 1 at time 2 (63.8061 M/hr). Boxes on this rank: 1 / 83
GRAMRLevel::advance level 2 at time 2 (63.7841 M/hr). Boxes on this rank: 1 / 83
GRAMRLevel::advance level 3 at time 2 (63.7631 M/hr). Boxes on this rank: 1 / 32
GRAMRLevel::advance level 4 at time 2 (63.7424 M/hr). Boxes on this rank: 1 / 32
GRAMRLevel::advance level 5 at time 2 (63.7217 M/hr). Boxes on this rank: 1 / 48
GRAMRLevel::advance level 6 at time 2 (63.2988 M/hr). Boxes on this rank: 1 / 80
GRAMRLevel::advance level 7 at time 2 (63.2653 M/hr). Boxes on this rank: 2 / 216
GRAMRLevel::advance level 8 at time 2 (63.2115 M/hr). Boxes on this rank: 3 / 345
GRAMRLevel::advance level 9 at time 2 (63.1339 M/hr). Boxes on this rank: 4 / 392
GRAMRLevel::advance level 9 at time 2.00391 (63.1969 M/hr). Boxes on this rank: 4 / 392
GRAMRLevel::advance level 8 at time 2.00781 (63.2659 M/hr). Boxes on this rank: 3 / 345
GRAMRLevel::advance level 9 at time 2.00781 (63.1959 M/hr). Boxes on this rank: 4 / 392
GRAMRLevel::advance level 9 at time 2.01172 (63.2626 M/hr). Boxes on this rank: 4 / 392
GRAMRLevel::advance level 7 at time 2.01562 (63.3282 M/hr). Boxes on this rank: 2 / 216
-UU-:----F1  pout.0          7% (1082,27)  (Text Fill) ----------------------------------
```

Rank 0

```
GRAMRLevel::advance level 2 at time 0 (0 M/hr). Boxes on this rank: 0 / 83
GRAMRLevel::advance level 3 at time 0 (0 M/hr). Boxes on this rank: 0 / 32
GRAMRLevel::advance level 4 at time 0 (0 M/hr). Boxes on this rank: 0 / 32
GRAMRLevel::advance level 5 at time 0 (0 M/hr). Boxes on this rank: 0 / 48
GRAMRLevel::advance level 6 at time 0 (0 M/hr). Boxes on this rank: 0 / 80
GRAMRLevel::advance level 7 at time 0 (0 M/hr). Boxes on this rank: 1 / 216
GRAMRLevel::advance level 8 at time 0 (0 M/hr). Boxes on this rank: 1 / 386
GRAMRLevel::advance level 9 at time 0 (0 M/hr). Boxes on this rank: 1 / 392
GRAMRLevel::regrid level 9 at time 0.5 (68.3608 M/hr). Boxes on this rank: 1 / 392
GRAMRLevel::regrid level 8 at time 1 (87.5216 M/hr). Boxes on this rank: 1 / 385
GRAMRLevel::regrid level 9 at time 1 (87.0986 M/hr). Boxes on this rank: 2 / 404
GRAMRLevel::regrid level 9 at time 1.5 (92.9475 M/hr). Boxes on this rank: 1 / 355
-UU-:----F1  pout.199        37% (72,27)    (Text Fill) ----------------------------------
```

Rank 199

1 box on rank 0 and level 0

Total # boxes on this level

- Check pout.0:
  - too many boxes?
- pout.last:
  - too many empty?
- Aim for all ranks to have similar boxes to compute
- Finer levels more important to be well balanced

Load balancing

Some possible problems:

- Too many/too few boxes..

What can you do?

- Change #nodes
- Chance #ranks/node, #threads/rank (make sure they still multiply to #cpus/node for the system)

- Very low M/hr even if boxes are okay?

- Usually can start with 1 cpu per process (better MPI parallelisation)
- May not have enough memory on 1 cpu -> try increasing #threads
- Keeping #nodes ~ #boxes * (cpus per process)/(cores per node)

Main points

- Make sure N_max is a multiple (at least double) of box_size
- Count #boxes on coarsest level -> ~ as many ranks is good place to start
- Start with 1-2 OpenMP threads
- Check pout.0 and pout.last files:
  - Too many/too few boxes? -> more/less #nodes or #ranks per node
  - What is M/hr after a few steps -> more threads per rank
- Any other tips?