OAK RIDGE National Laboratory

Modeling pre-Exascale AMR Parallel I/O Workloads via Proxy Applications

William F Godoy[1], Jenna Delozier[2], Gregory R Watson[1]

[1]Computer Science and Mathematics Division, Oak Ridge National Laboratory

[2]College of Computing, Georgia Institute of Technology

Prepared for iWAPT 2022 workshop, in conjunction with IEEE IPDPS 2022

U.S. DEPARTMENT OF ENERGY

022

Content

- Introduction: AMR outputs and characterization
- Understanding AMReX Castro parallel outputs
- Proxy App: MACSio modeling formulation and results
- Summary and Future plans

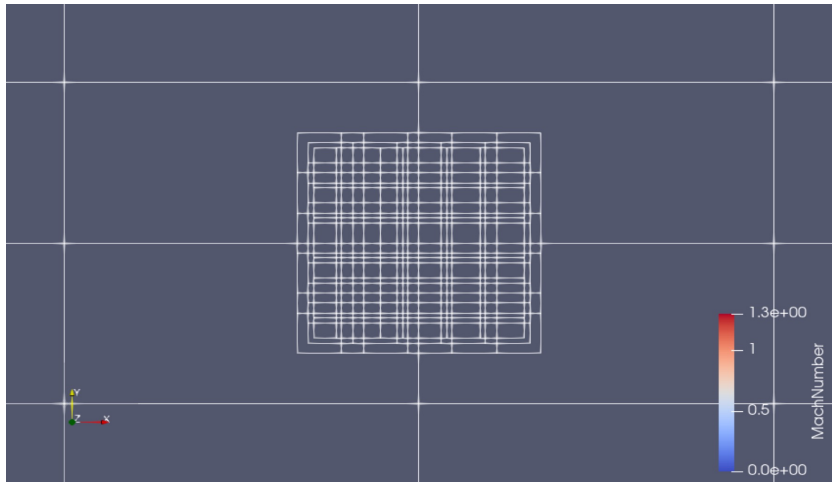
Introduction: AMR outputs

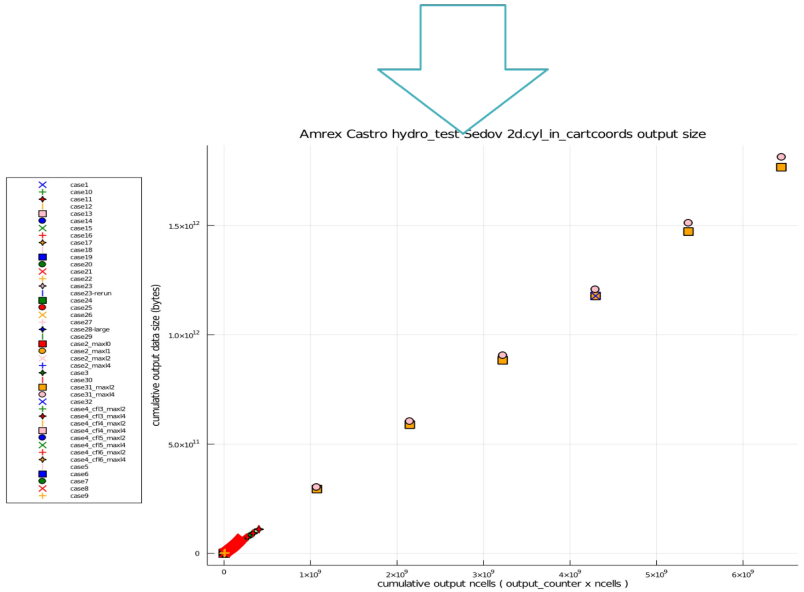
- Adaptive Mesh Refinement (AMR) is a powerful technique for solving partial differential equations (PDEs)
- Depending on user needs, it can cross between compute to I/O bound intensive simulations
- I/O patterns depend on several variables in the meshing, solution, partitioning
- I/O is becoming a bigger bottleneck as we enter the exascale era

Introduction: AMR IO Characterization

- **Understanding I/O characteristics of Adaptive Mesh Refinement (AMR) Input/Output Patterns**
- **“Predictability” in pre-Exascale systems (Summit) via proxy applications.**
- **Cost of Analysis Data and Checkpoint-Restart**
- **Better decision making for AMR codes**
  - **When do I use Burst Buffers or the File System?**
  - **How often do I write data? Checkpoint? Analysis?**
  - **What’s the I/O cost overhead in my application?**
  - **How much “in situ” data can I process?**
  - **What format or backend should I use?**

GOAL: “Develop adaptable modeling and proxy applications frameworks to characterize AMR code I/O for better decision making in the upcoming Exascale era”

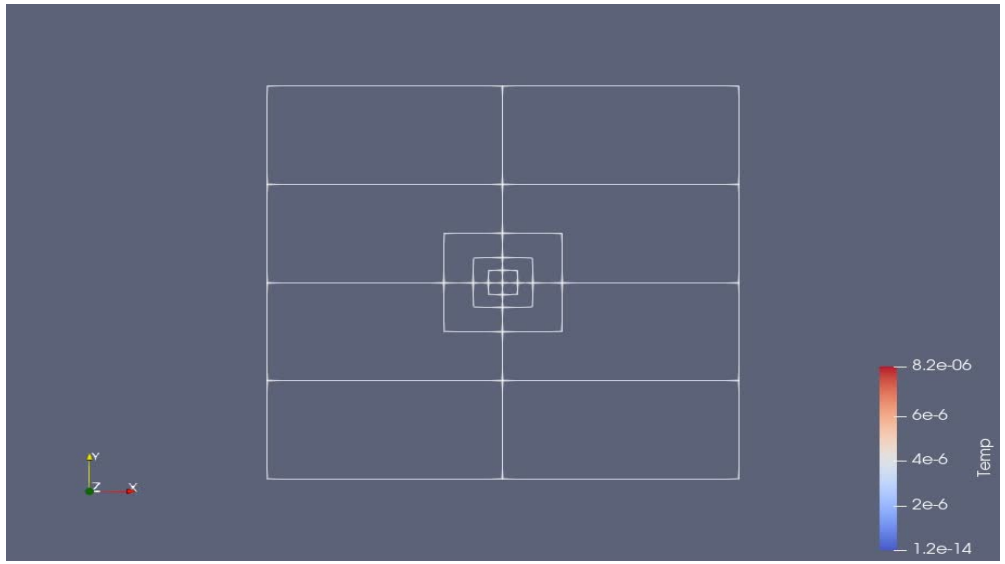


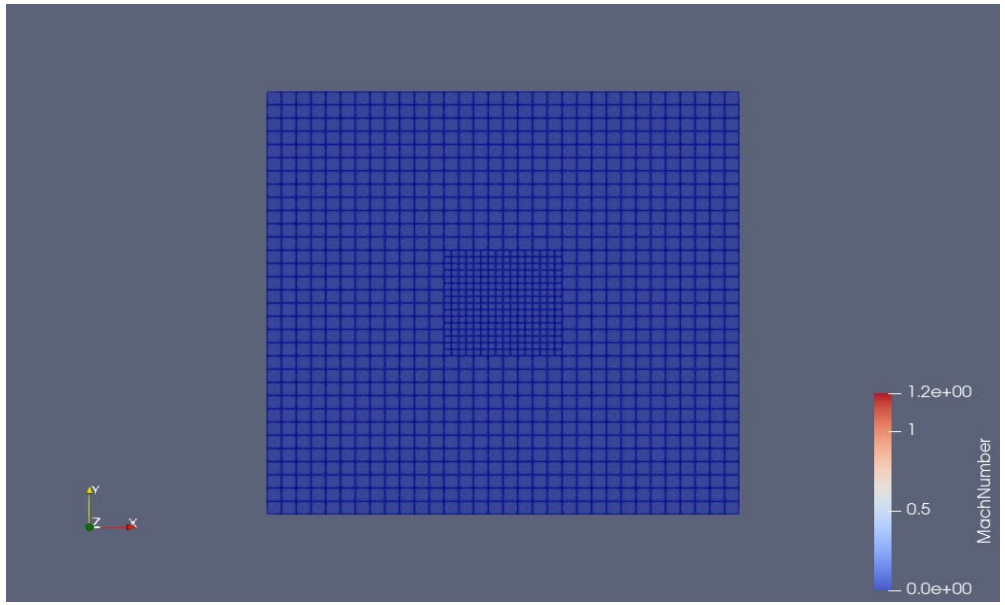


OAK RIDGE National Laboratory

Understanding AMReX Castro parallel outputs

- **Output production rate depends on**
  - **physical and simulation parameters**
  - **mesh generation**
  - **parallel partition**
- **Not easily predictable**
- **Let's look at a simple Sedov hydrodynamic test**
- **Symmetric, 2D Mach number using 3 levels**





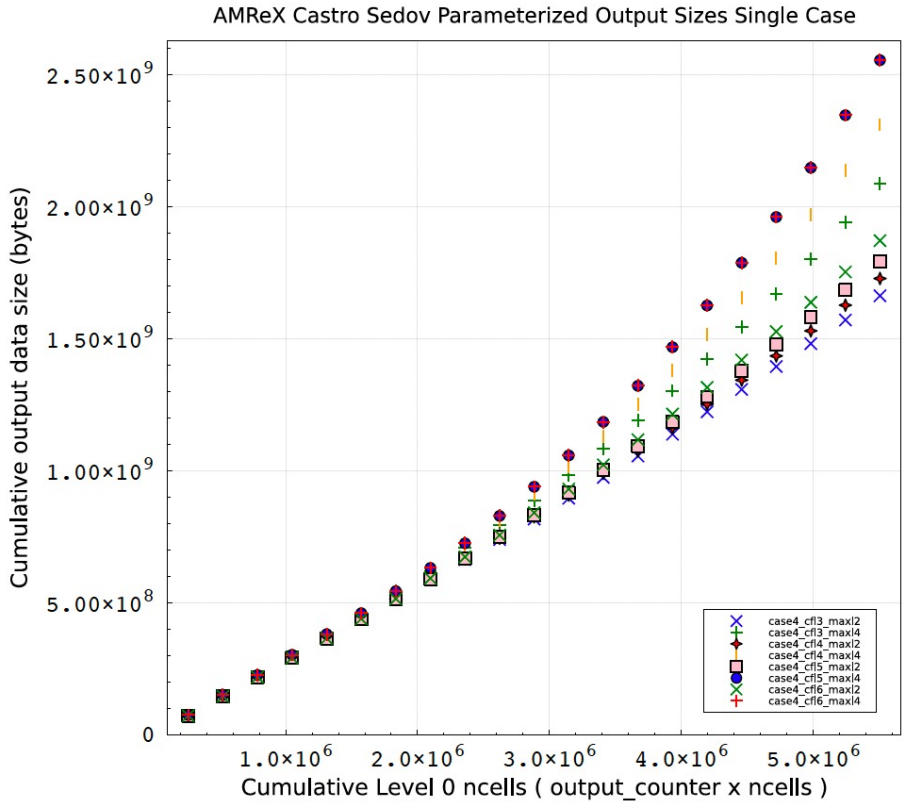
Understanding AMReX Castro parallel outputs

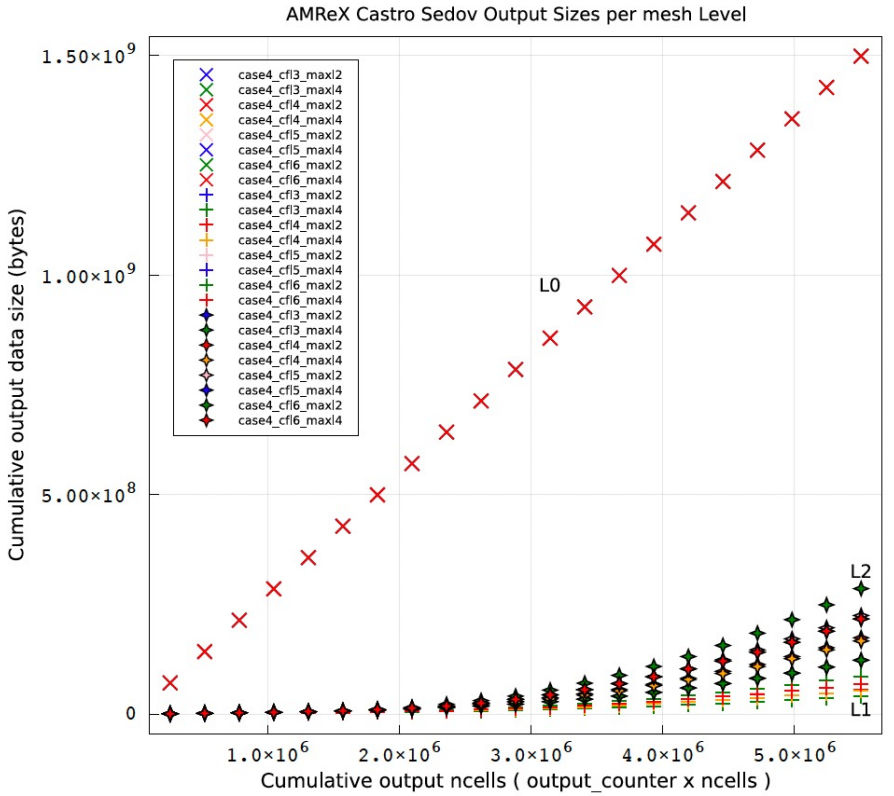
- **AMReX Castro https://amrex-astro.github.io/Castro/ produces analysis data on a <Timestep, Level, Rank> basis**

```
AMReX Castro Simulation Output
├── sedov_2d_cyl_in_cart_plt00000 (step directory - one per plot interval)
│   ├── Header
│   ├── job_info
│   ├── Level_0 (level directory - one per level)
│   │   ├── Cell_D_00000 (cell data file - one per rank)
│   │   ├── Cell_D_00001
│   │   ├── ...
│   │   ├── Cell_D_0000N (N = number of MPI processes)
│   │   └── Cell_H (mesh metadata file)
│   ├── Level_1
│   ├── Level_2
│   ├── ..
│   └── Level_L
├── sedov_2d_cyl_in_cart_plt00020
├── ..
└── sedov_2d_cyl_in_cart_pltMMMMM (final step output)
```

Understanding AMReX Castro parallel outputs

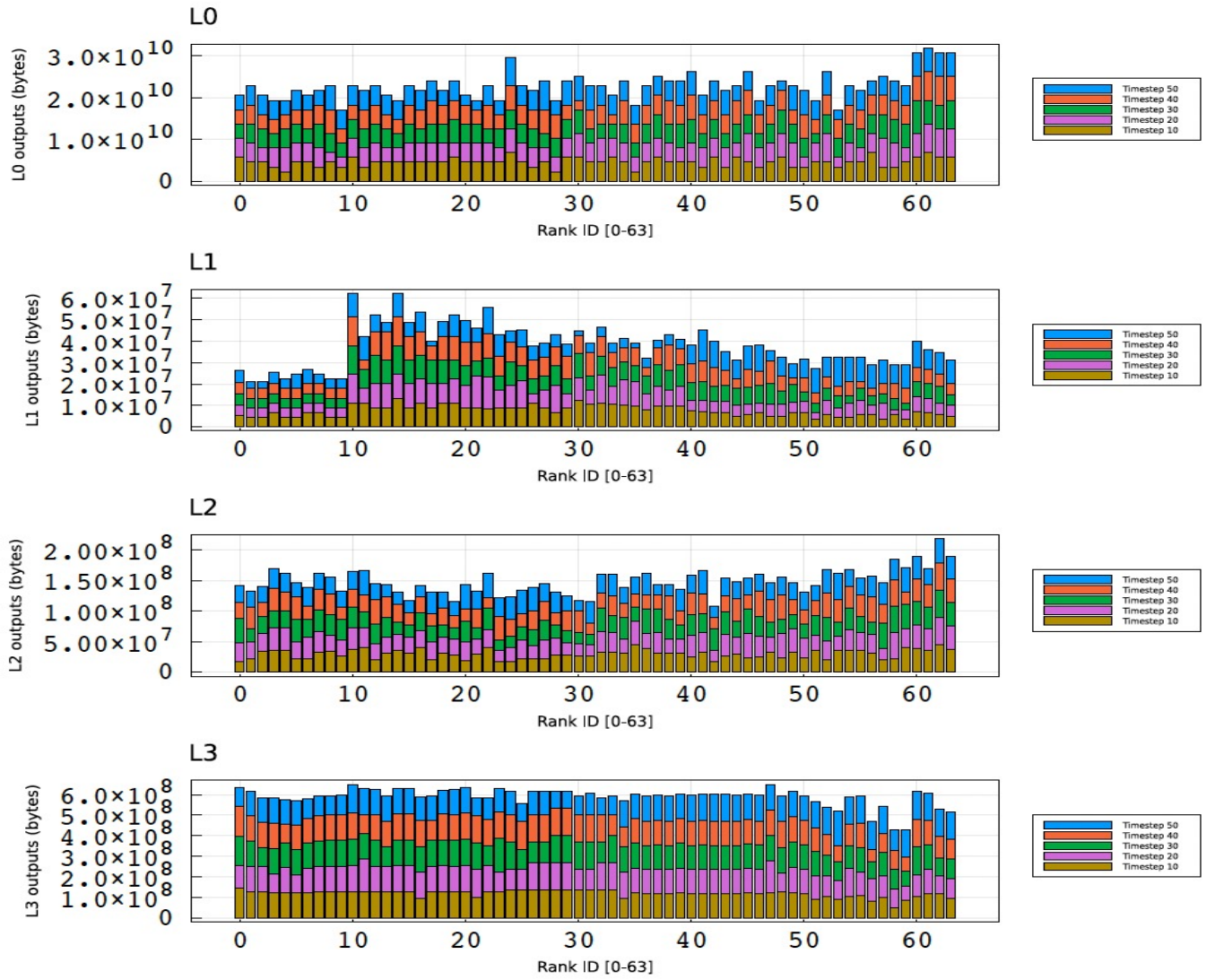
- **<Timestep, Level> → "smooth production"**
- **<Rank> → "unpredictable production"**





Understanding AMReX Castro parallel outputs

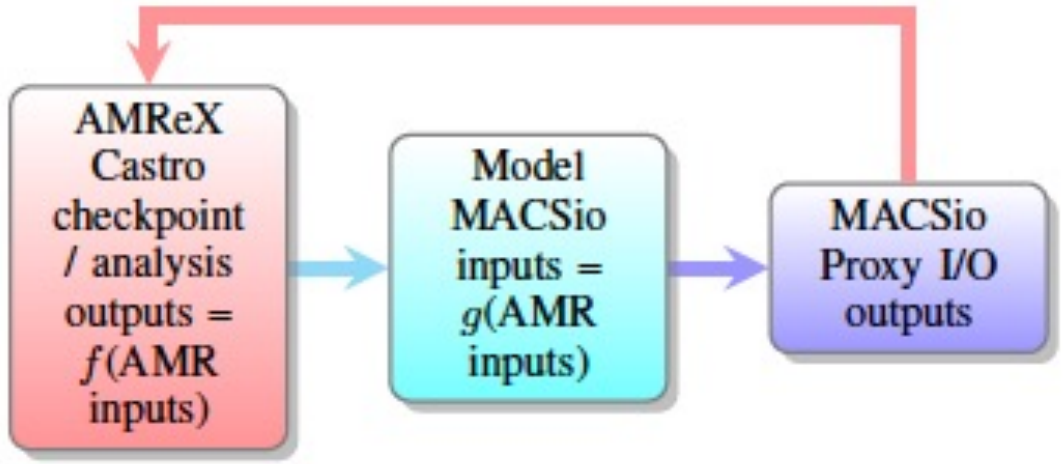
- **<Rank> → "unpredictable production"**



Proxy App: MACSio modeling formulation

- Generate cases on Summit, AMReX Castro to do a characterization study. [Outputs] = f([Inputs])
  - Inputs: problem size, output frequency, partition, **AMR levels, CFL condition**
  - Outputs: analysis data sizes
- MACSio https://github.com/LLNL/MACSio is a simple MPI based executable that produces "Kernel" outputs via arguments. <Step, Rank> production

```
$ mpirun -np 32 macsio --interface  default --avg_num_parts 8 --part_size 100K --parallel_file_mode MIF 32
```




```
MACSio data output
├─ data
│  └─ macsio_json_{taskID}_{stepID}.json
│     ├─ macsio_json_00000_000.json
│     ├─ macsio_json_00000_001.json
│     ├─ ...
│     ├─ macsio_json_00000_{nsteps}.json
│     ├─ ...
│     └─ macsio_json_{nprocs}_{nsteps}.json
└─ metadata
   └─ macsio_json_root_{stepID}.json
      ├─ macsio_json_root_000.json
      ├─ macsio_json_root_001.json
      ├─ ...
      └─ macsio_json_root_{nsteps}.json
```

Proxy App: MACSio modeling formulation

- Generate cases on Summit, AMReX Castro to do a characterization study. [Outputs] = f([Inputs])
  - Inputs: problem size, output frequency, partition, **AMR levels, CFL condition**
  - **Understand parameters that drive I/O**

Listing 2: Input configuration file for the AMReX-Astro Castro Sedov inputs.2d.cyl_in_cartcoords case.

```
# INPUTS TO MAIN PROGRAM
max_step = 500
stop_time = 0.1

# PROBLEM SIZE & GEOMETRY
geometry.is_periodic =  0 0
geometry.coord_sys   =  0  # 0 => cart
geometry.prob_lo     =  0    0
geometry.prob_hi     =  1    1
amr.n_cell           = 32   32

# BC FLAGS
# 0 = Interior 3 = Symmetry
# 1 = Inflow   4 = SlipWall
# 2 = Outflow  5 = NoSlipWall
castro.lo_bc       =  2   2
castro.hi_bc       =  2   2

# WHICH PHYSICS
castro.do_hydro = 1
castro.do_react = 0

# TIME STEP CONTROL
# CFL number for hyperbolic system
castro.cfl            = 0.5
# scale back initial timestep
castro.init_shrink    = 0.01
# maximum increase in dt
# over successive steps
castro.change_max = 1.1

# DIAGNOSTICS & VERBOSITY
# timesteps between computing mass
castro.sum_interval = 1
```

Parameter	Range
amr.max_step	40 - 1000
amr.n_cell	(32×32) - $(131,072 \times 131,072)$
amr.max_level	2 - 4 (1 to 3 levels)
amr.plot_int	1 - 20
castro.cfl	0.3 - 0.6
nprocs	1 - 1,024
Summit nodes	1 - 512 (1/9 total system)

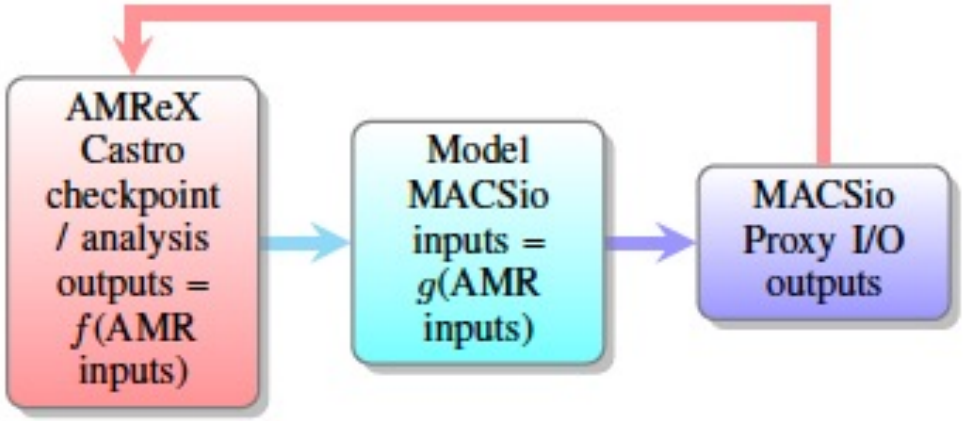
TABLE III: AMReX Castro input configuration file parameters range for the Sedov case running imulations to produce different output sizes.

Proxy App: MACSio modeling formulation

- Relate MACSio inputs to AMReX-Castro inputs

MACSio Argument	Description
`interface`	output type hdf5, json (miftmpl), silo
`parallel_file_mode`	File Mode: multiple independent, single
`num_dumps`	number of dumps to marshal (buffer)
`part_size`	per-task mesh part size
`avg_num_parts`	average number of mesh parts per task
`vars_per_part`	number of mesh variables on each part
`compute_time`	rough time between dumps
`meta_size`	additional metadata size per task
`dataset_growth`	multiplier factor for data growth

TABLE II: MACSio command line arguments used to model AMReX-Castro outputs.



Functional form:

```
jsrun -n nproc
macsio
 --interface miftmpl
 --parallel_file_mode MIF nproc
 --num_dumps amr.max_steps / amr.plot_int
 --part_size f(amr.n_cell)
 --avg_num_parts 1
 --vars_per_part 1
 --compute_time f(platform,all_inputs)
 --meta_size f(all_inputs)
 --dataset_growth f(amr.n_cell,castro.cfl,amr.max_level,...)
```

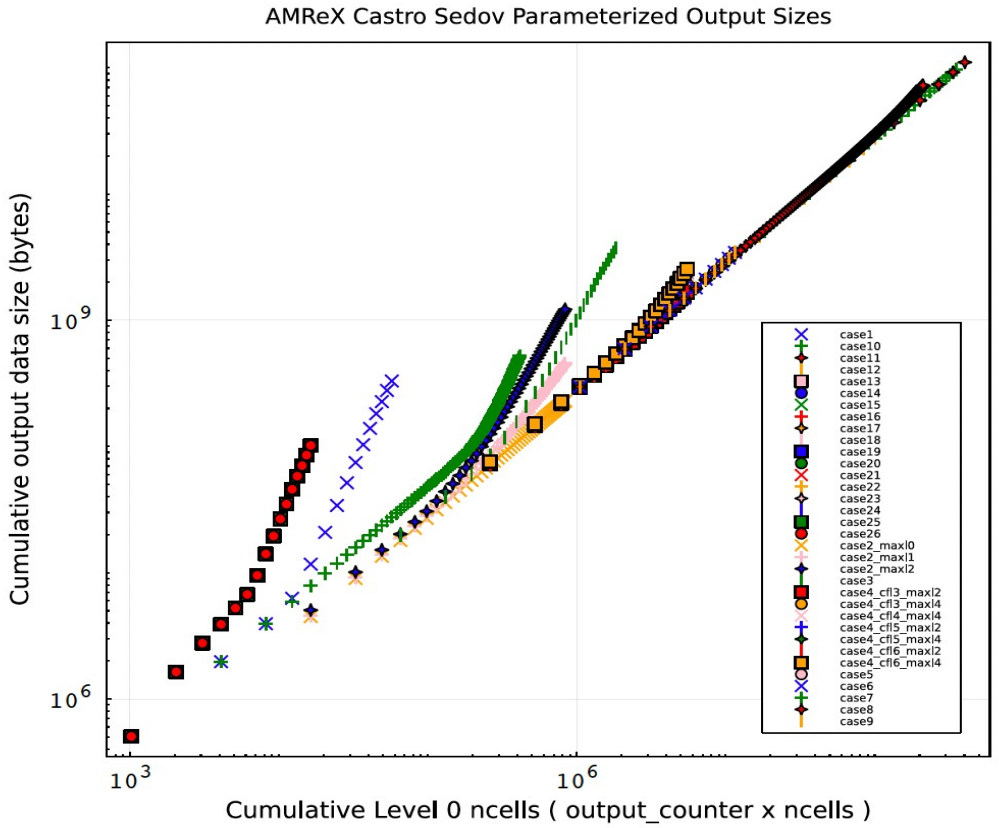
$$\texttt{part\_size} = f\frac{8\,Nx\,Ny}{nprocs}\ [bytes]$$

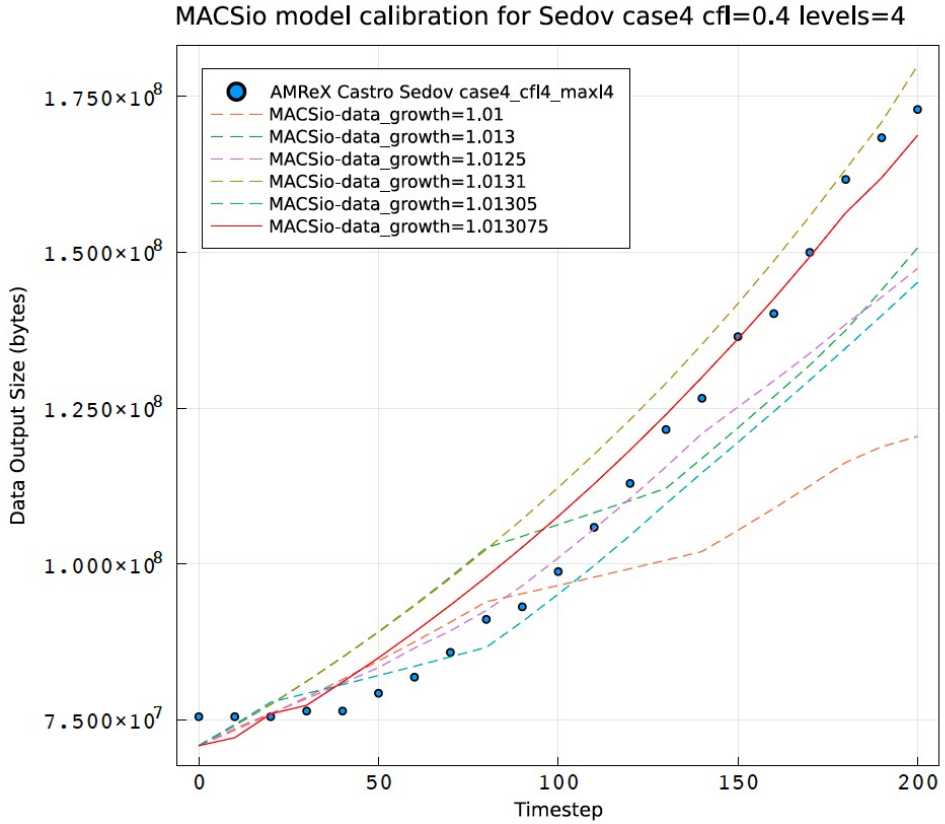
$$f \approx [23 - 25]$$

Proxy App: MACSio modeling results

- Run the AMReX-Castro Sedov baseline case over several configurations on the Summit supercomputer
- Match MACSio model with Sedov output sizes until convergence
- Validate the model over several Sedov configurations

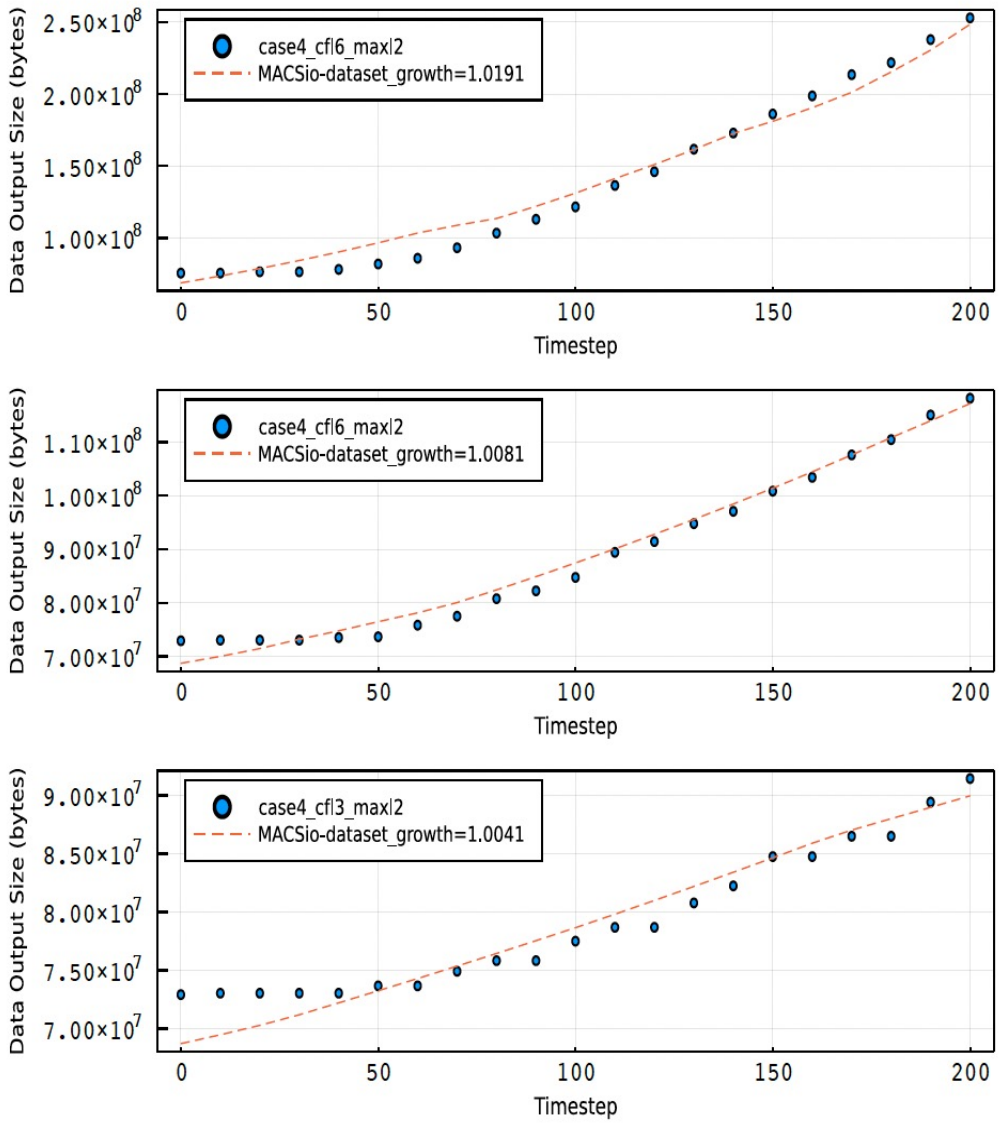
$$\texttt{part\_size} = f\frac{8\,Nx\,Ny}{nprocs}\ [bytes]$$
$$f \approx [23 - 25]$$





Proxy App: MACSio modeling results

- Validate over different cases to model overall "Timestep" output to a certain degree of confidence



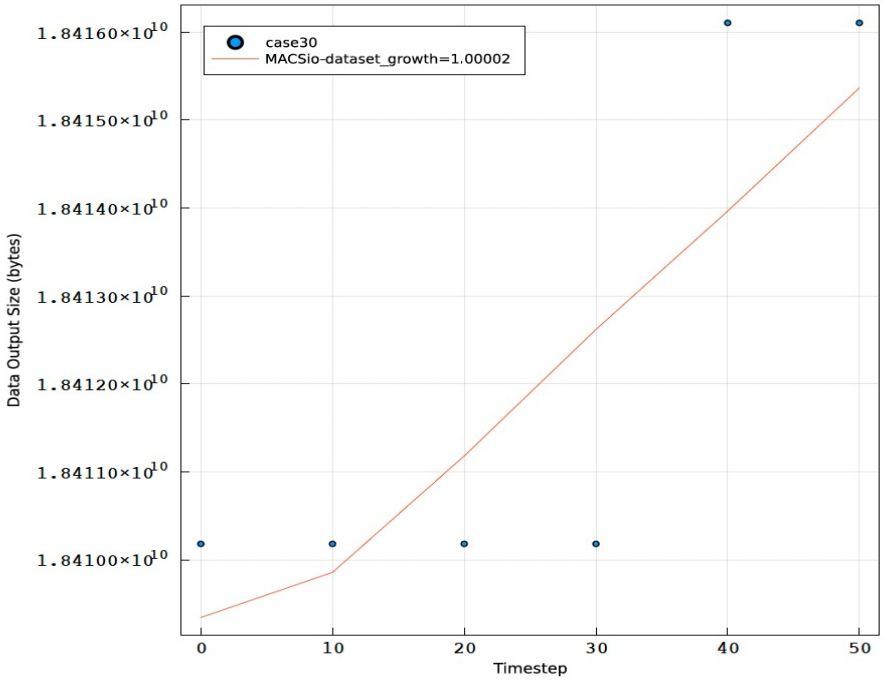


Fig. 11: Comparison of a large L0 mesh $= 8192 \times 8192$ Sedov non-smooth simulation output against the proposed MACSio kernel model.

Summary and Future Work

- We present a simple model formulation for understanding Adaptive Mesh Refinement data outputs via a proxy application like MACSio
- Empirical models can encapsulate the unpredictable per-rank parallel I/O rates up to model AMR levels and timestep data output rates for a range of problem sizes on the Sedov hydrodynamics case
- Future Research Questions:
  - Can we extend this formulation for other AMR configurations?
  - Do we need better granularity in the proxy app and other backends (HDF5, ADIOS2)?
  - Can proxy based modeling help understand exascale AMR I/O bottlenecks?
  - Can we provide autotune system policy for AMR applications at scale?

Acknowledgements

This research was supported by the Exascale Computing Project (17-SC-20-SC), a joint project of the U.S. Department of Energy's Office of Science and National Nuclear Security Administration, responsible for delivering a capable exascale ecosystem, including software, applications, and hardware technology, to support the nation's exascale computing imperative.

This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

iWAPT 2022

Thanks!