Optimizing I/O performance for AMR Code: A case study with RAMSES in Astrophysics

By Loïc STRAFELLA
Table of contents

I. Introduction
   • Astrophysics and AMR

II. I/O Optimization
   • I/O bottlenecks in RAMSES
   • Strategy of optimization

III. The impact of the data model
   • Specific AMR TB data model

IV. Post-processing strategy and I/O optimization
   • “Region-of-Interest”, “Level-of-detail”
Part I

Why astrophysicists need AMR?
Introduction

The universe (~ giga/mega parsec):
- Filamentary structure: “cosmic web”

Galaxy (~ kilo parsec):
- Classification: elliptical, irregular, spiral, ...
- Contains: dark matter, gas, dust
- Different thermodynamic states for gas:
  - Hot ionized (HIM), warm ionized (WIM), warm/cold neutral (WNM, CNM), molecular

Molecular cloud (~ tens parsec):
- Made of molecular gas: H$_2$, CO, ...
- Found in spirals arms, central bar, ...
- Spot for stellar formation:
  - Kennicutt-Schmidt law [Kennicutt Jr, 1998]

1 parsec ~ $3.10^{13}$ km
Introduction

Multi-scale processes:

➢ Spatial resolution vary by several order of magnitude:
  ● Example: galactic scale ~30 kpc, molecular cloud 50 pc, ...
➢ Field magnitude can also vary by several order of magnitude:
  ● Density $10^{-6} \text{H/cm}^3$ to $10^{+6} \text{H/cm}^3$

Multi-physics:

➢ Large scale gravitational/hydrodynamical instabilities
➢ Thermodynamics, magnetic fields, ...
➢ Star formation models and its feedback (ionization, supernovae, ...)

Need of numerical simulation with specific mesh strategy (AMR)

RAMSES [Teyssier, 2002]:

➢ MPI only, Fortran
➢ Cell-based
➢ Fully threaded tree [Khokhlov, 1998]

AMR: Adaptive Mesh Refinement
Introduction: objectives and limitations

State-of-the-art simulations:

- **Sub-galactic region**, e.g. [Brucy et al, 2020], [Colman et al, 2022]
  - Missing large scale turbulence injection mechanism

- **Simplified physics models**
  - No heating/cooling process, [Renaud et al, 2013]
  - No star formation (SF), no feedback [Fensch et al. 2023]

- **Short physical time evolution at high resolution**

- **High resolution only in high density region**
  - Missing instabilities that may develop in low density region

Requires preliminary work to:

- Reduce the computational cost
- Improve code scalability in terms of consumed memory
- **Improve code scalability regarding I/O**
Objective:
Study the regulation of star formation in a Milky Way like galaxy

Data Management Plan (DMP): FAIR – open science « Findable, Accessible, Interoperable, Reusable »

DMP for the simulation Exa-Milkyway:
• Code **RAMSES** [Teyssier, 2002] (MPI only – no OpenMP)
• **Supercomputer**: CPU AMD Epyc 128 cores / node, 256 Go, Irène **TGCC** *
• **Target**: 16.384 – 50.000 MPI processes
• **57 Millions** CPU-hours, **GENCI** **
• 90 To of disk space: 50 To checkpoint + 40 To for analysis
• Open science

* **TGCC**: Très Grand Centre de Calcul du CEA
** **GENCI**: Grand équipement national de calcul intensif
**Bottlenecks and critical points:**

- **Low I/O scalability:** ~ 8,000 MPI processes
  - Time to write/read data (checkpoints): ~ 1h, [Renaud, 2013]
  - 10 to 20% of time allocation spent on I/O
- **High number of files:**
  - «file-per-process» strategy
  - Limited quota: 2 millions* (~30 snapshots** max.)
- **Complex I/O pattern** due to AMR [Wautelet and Kestener, 2011]
- **Only one data flow**
  - Data used for both analysis and checkpoint
- **High data volume production**
  - Limited quota: 100 To* (~30 snapshots max.)
- **Low scalability memory footprint**
- **Not possible** to change the number of MPI processes at restart

* Quota per project at TGCC
** Snapshot: directory / files associated to a given time
After I/O optimization

**Solutions:**

- **Improved I/O scalability up to 50,000 MPI**
  - Time to write or read data (16.384): ~ **3 minutes** (4,7 To) (vs ~ 1h)

- **Reduced number of files:**
  - ~200,000 files : ~ **150 snapshots** (vs ~ 30)

- Separation of data flow, new data model, data compression
  - 90 To disk usage: **140 snapshots for analysis, 10 snapshots checkpoints**
  - **200 Go / snapshot** (analysis) vs **4,7 To / snapshot** (checkpoint)

- **Optimization of memory footprint***:
  - **Reduction of 64 Go / node** (25 % of available memory per node)

- New feature: changing the number of MPI processes on restart

**Tested, validated and used** in production run Exa-MilkyWay

---

* With 16384 MPI on the ExaMilkyWay simulation

** Patch "LIGHT_MPI_COMM" open sourced for the community
Part II

I/O Optimization
I/O: Some definitions

Different structures:

- **Computational data structure**: linked list, hash map, ....
- **Data model (for I/O)**: “unstructured”, “custom”*, “lightAMR”, ...
- **Data format**: compressed or not, specific encoding strategy, ...
- **File format (and file structure)**: linked to the I/O library
  - Different file format and file structure possible for a given data model

* “custom” = specific to a given code
I/O: RAMSES I/O strategy

Initial state:
- « file-per-process » strategy
  - *Lot of files*: does not take advantage of parallel filesystem
- Custom data model
  - *Per level data write/read*: lot of small I/O calls
- Custom file format with Posix interface
  - Nightmare for post-processing tools
- Single data flow for multiple purposes: checkpoint & analysis
  - Too much data for analysis
  - Unnecessary computation for checkpoint/restart
  - Use of temporary arrays

How to optimize the I/O?
- Try to optimize file system or user parameters
- Use a specific library (made by expert in the field)
- Small number of I/O calls with large amount of data
- Reduce the data volume (data model, data compression, reduce precision, …)
I/O: Optimization strategy

Objectives:
- Change I/O strategy ("file-per-process")
- Use of specific parallel I/O library
- Separation of data flow based on purpose
- Change I/O pattern

- Hercule parallel I/O library
- Split current data flow
- New data models

*: Developed at CEA-DAM, [Bressand et al, 2012]
**I/O: Optimization strategy**

**Objectives:**
- Change I/O strategy (“file-per-process”)
- Use of specific parallel I/O library
- Separation of data flow based on purpose
- Change I/O pattern

**Parallel I/O libraries:** HDF5, NetCDF, MPI-IO, **Hercule** *, ...

**Advantages** of Hercule:
- Native parallel I/O based on application parallelism (MPI)
- Simple API (compare to HDF5 for example)
- Database like strategy
- “Key – value”: easy access to data

**Separation of data flow:**
- One specific for checkpoint/restart
- One specific for analysis

*: Developed at CEA-DAM, [Bressand et al, 2012]
I/O: Hercule

Key points:
- I/O strategy: «multiple-shared-file»
- Database like access: based on time, on MPI rank (domain), ...
- Data accumulation in the same file (“append”)
I/O: Hercule database

Two different Hercule databases:

HProt  ➔ For checkpoint/restart, no specific semantic, “basic” write/read operation
HDep  ➔ For analysis and post-processing, specific data model and semantic API
I/O: New data flow diagram

New RAMSES data flow with Hercule's databases

- **HDep**: High “dump” frequency, low volume
- **HProt**: Low “dump” frequency, high volume
Benchmarks Write: RAMSES + HERCULE

Benchmarks:

- **Checkpoint/restart** Sedov 3D use case – $2048^3$ cells – I/O only (no time integration) – 500 Go
- **Hercule**: Number of Contributor Per File (« NCF »)

![Chart showing output capacity GB/s for different numbers of cores and files for RAMSES and HERCULE with NCF 4, 8, and 16.]

Number of files:
- 16.384
- 2.048
- 1.024
- 512

CEA TGCC - partition « scratch »
Part III

A new data model for TB-AMR
LightAMR data model

Why we need a new data model?
- No standard data model for tree-based AMR
- RAMSES data model not specific for post-processing
- High data volume produced with other data model
  - « unstructured » not adapted for AMR data
- Need to keep the hierarchy of cells

New LightAMR post-processing specific data model

How to make it a “standard”? (Open science)
- Provide clear documentation (scientific publication, “readthedocs”, ...)
- Independent from the code that produce the data
- Self-consistent
- Example of usage / extraction of information (Optionnel)

*“LightAMR format standard and lossless compression algorithms for AMR: RAMSES use case”, Strafella & Chapon, 2022
Requirements for the new data model:

- **Compact** representation of the mesh
  - Reduce the cost of storage or transfer
  - Keep implicit as much as info. as possible, e.g. : cell centers
- Keep the hierarchy of cells
  - Allow optimization in post-processing: level-of-details, region of interest, ...
LightAMR data model & additional algorithms

3 additional algorithms:

- **Redundancy removal** (« Tree pruning »):
  - Remove ghost cells and nodes
  - Small computational and memory cost
  - **High impact on data volume**: 8 - 45 % (depending on use case)

- **Mesh data compression**:
  - Use of « run-length-encoding »
  - High data compression ratio
  - Mesh description negligible: < 1 % of total volume

- **Floating point data compression** (32 or 64 bits):
  - “delta” compression, “parent-child-predictor” (PCP)
  - Small memory cost, fast (~ Go/s)
  - **Lossless**: no loss of information (bit to bit), **low ratio**
  - **Lossy**: max. abs. Point-wise error + centered around 0, **high ratio**
LightAMR: data volume reduction

- Test dataset (real simulation): FRIGG\(^{(1)}\), ORION\(^{(2)}\), Extreme-Horizon\(^{(3)}\), Cosmic-Dawn III\(^{(4)}\)

<table>
<thead>
<tr>
<th>RAMSES</th>
<th>LightAMR ((E_{rel} \leq 2e^{-3}))</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AMR</td>
</tr>
<tr>
<td>ORION</td>
<td>9 Go</td>
</tr>
<tr>
<td>FRIGG</td>
<td>30 Go</td>
</tr>
<tr>
<td>Extreme-Horizon</td>
<td>500 Go</td>
</tr>
<tr>
<td>CoDa III</td>
<td>16 To</td>
</tr>
</tbody>
</table>

- From production Exa-MilkyWay (RAMSES + LightAMR + Hercule)

<table>
<thead>
<tr>
<th>Protection/reprise</th>
<th>LightAMR ((E_{rel} \leq 2e^{-4}))</th>
<th>Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exa-MilkyWay</td>
<td>4,7 To</td>
<td>x24</td>
</tr>
</tbody>
</table>

1. [P. Hennebelle. 2018], conversion to LightAMR
2. [E. Ntormousi and P. Hennebelle., 2019], conversion to LightAMR
3. [S. Chabanier et al., 2020], conversion to LightAMR
4. [L. Joseph et al., 2022], conversion to LightAMR

* lossless compression
** lossy compression
Part IV

Post-processing LightAMR data
Méthodologie de post-traitement

Data for analysis:
- Used data model: **LightAMR**
- Number of domains
  - 512 (**Orion**) up to 131,072 (**CosmicDawn III**)
- Number of cells
  - ~500 millions (**Orion**) to ~500 billions (**CosmicDawn III**)
- Data volume
  - From Gigabytes to Terabytes (**LightAMR** compressed or not)

Requirements for a post-processing tool:
- Good I/O scalability
  - Compatible with parallel I/O library: Hercule, HDF5
  - Need of **MPI**
- Good computational scalability
- Compatible with multicores architectures
  - Shared memory parallelism: **OpenMP**
Strategy for post-processing

Take advantage of Hercule “per domain” data access:

- Hybrid parallelism: distributed memory (MPI) and shared memory (OpenMP)
  - 1 MPI — a list of domains
  - 1 openMP thread — one domain
Strategy for post-processing

Different optimization strategy:

- "Region of Interest" (ROI)
  - Select domains of interest: optimize I/O and memory
  - Select cells to analyze: optimize computation and memory
- "Level-of-details" (LOD)
Conclusion
Strategy for post-processing

More work to do:

- More I/O benchmarks: checkpoints vs analysis, impact of compression, …
- Extension of LightAMR for other AMR strategy

Recommendation for codes with I/O bottleneck and/or high data volume

- Check the data model
- Use of data compression?
- Double precision? Single precision?
- Field selection?
- Which I/O library?
Workshop EXA-DI \textbf{Block-structured AMR} @\textit{Exascale}

Questions ?

Loic Strafella
Data pipeline

Development and optimization of the data pipeline
- **Parallel I/O**: Ramses + Hercule, HDF5 extension
- **Data model**: LightAMR
- **Data compression**: algorithms CPS52, PCP
- **HPDA**: post-processing tool, python interface
Schéma d'un système **Lustre**

- **Noeuds de calculs** : application
- **Serveur de méta-données** : information sur les dossiers et fichiers
- **Serveur de stockage** : stockage des données brutes
- **Réseaux**
LightAMR data model

Mesh description:

Refinement
\[
\begin{cases}
1 & : \text{refined (node)} \\
0 & : \text{not refined (leaf)}
\end{cases}
\]

Ownership
\[
\begin{cases}
0 & : \text{owned} \\
1 & : \text{not owned}
\end{cases}
\]
LightAMR data model

Cell values:

- **Scalar field**: same order as node/leaf description (1:1 « mapping »)
  - Expecting a value on each **node**
- **Vector field**: split in **n** scalar fields (optimization in post-processing)
LightAMR: réduction de volume de données

**Extreme-Horizon**: 2.3 To **Hydro**. data + 490 Go of **AMR** (RAMSES)

![Graph showing data volume reduction comparison between RAMSES and lightAMR](image)

- **Modèle LightAMR**
- **Elagage**
- **Compression flottante sans perte**
- **Compression flottante avec perte**
Méthodologie de post-traitement

Utilisation de Région d'intérêt (ROI, « Region of Interest »)

➢ Sélection des domaines à traiter : optimisation des E/S + mémoire
➢ Sélection des cellules à traiter : optimisation des calculs + mémoire

Pavage Cubique Minimal

➢ Calculé à partir des méta-informations (clés de Hilbert) ou de la description du maillage
➢ Utilisé comme structure d'accélération pour le lancer de rayons
Fonctionnalité de l’outil d’analyse

Traitements et algorithmes disponibles :

- Agglomération de domaines
- Conversion: vtkHyperTreeGrid, non structuré, ...
- Analyse par secteurs, calcul de grandeurs dérivées, particules, ...
- Production de carte 2D (flou gaussien, « raytracing »)
- Calcul d'histogramme 1D, 2D, ...
- « Batch processing » en cas de ressource mémoire faible

Quelques chiffres:

- 1 Carte 2D par lancer de rayons (100 Mpixels) : ~ 30 secondes
  - 72 MPI + 40 threads / MPI : 2880 coeurs
  - ~ 30 milliards de cellules

Stratégie efficace avec les données testées:
- Scalable et “memory friendly”
- Tire profit des E/S parallèle (Hercule)
- Dédié au LightAMR
Quel Impact de la compression avec perte ?

Estimate the impact on results is essential, e.g. climate data [Baker et al, 2016]

Define a data compression strategy adapted to the needs (DMP)

No impact on analysis or scientific results
2D Map, power spectra, histogram, etc.