Enzo-E / Cello

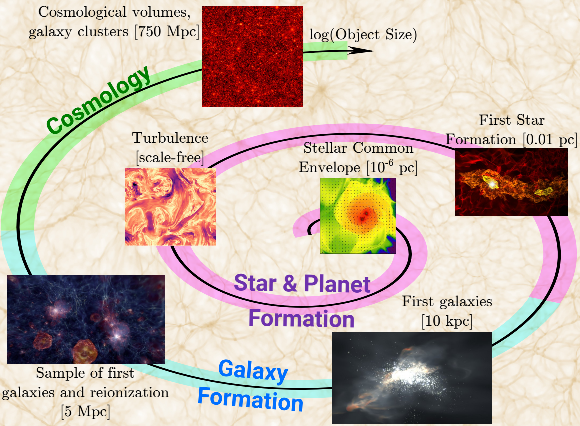
Adaptive Mesh Refinement Astrophysics using Charm++

James Bordner, Michael L. Norman

University of California, San Diego
San Diego Supercomputer Center

17th Annual Workshop on
Charm++ and Its Applications
2019-05-01/02

Scientific questions in astrophysics and cosmology



[John Wise]

The ENZO MPI-parallel astrophysics application

Applicable to a wide range of astrophysical and cosmological problems

Scientific Questions: *topics of interest*

• Star formation • Molecular cloud turbulence • Interstellar medium • Galaxy formation • Intergalactic medium • Galaxy clusters • Cosmic reionization . . .

Physics Equations: *mathematical models*

• Hydrodynamics (Euler equations) • Gravity ($\nabla^2\Phi = 4\pi G\rho$) • Chemistry/cooling • Magnetism • Radiation transport . . .

Numerical Methods: *approximate and solve*

• PPM, • ZEUS, • MUSCL • FFT • multigrid • Gadget cooling • Cloudy cooling • Grackle • Dedner MHD • MHD-CT • Implicit FLD • Moray . . .

Data Structures: *computer representation*

• Adaptive Mesh Refinement (SAMR) • Eulerian fields • Lagrangian particles . . .

The ENZO MPI-parallel astrophysics application

Supports many related physics capabilities

Scientific Questions: *topics of interest*

• Star formation • Molecular cloud turbulence • Interstellar medium • Galaxy formation • Intergalactic medium • Galaxy clusters • Cosmic reionization . . .

Physics Equations: *mathematical models*

• Hydrodynamics (Euler equations) • Gravity ($\nabla^2\Phi = 4\pi G\rho$) • Chemistry/cooling • Magnetism • Radiation transport . . .

Numerical Methods: *approximate and solve*

• PPM, • ZEUS, • MUSCL • FFT • multigrid • Gadget cooling • Cloudy cooling • Grackle • Dedner MHD • MHD-CT • Implicit FLD • Moray . . .

Data Structures: *computer representation*

• Adaptive Mesh Refinement (SAMR) • Eulerian fields • Lagrangian particles . . .

The ENZO MPI-parallel astrophysics application

Implements a variety of sophisticated numerical methods

Scientific Questions: *topics of interest*

• Star formation • Molecular cloud turbulence • Interstellar medium • Galaxy formation • Intergalactic medium • Galaxy clusters • Cosmic reionization . . .

Physics Equations: *mathematical models*

• Hydrodynamics (Euler equations) • Gravity ($\nabla^2\Phi = 4\pi G\rho$) • Chemistry/cooling • Magnetism • Radiation transport . . .

Numerical Methods: *approximate and solve*

• PPM, • ZEUS, • MUSCL • FFT • multigrid • Gadget cooling • Cloudy cooling • Grackle • Dedner MHD • MHD-CT • Implicit FLD • Moray . . .

Data Structures: *computer representation*

• Adaptive Mesh Refinement (SAMR) • Eulerian fields • Lagrangian particles . . .

The ENZO MPI-parallel astrophysics application

Enabled by adaptive mesh refinement with particles and fields

Scientific Questions: *topics of interest*

• Star formation • Molecular cloud turbulence • Interstellar medium • Galaxy formation • Intergalactic medium • Galaxy clusters • Cosmic reionization . . .

Physics Equations: *mathematical models*

• Hydrodynamics (Euler equations) • Gravity ($\nabla^2\Phi = 4\pi G\rho$) • Chemistry/cooling • Magnetism • Radiation transport . . .

Numerical Methods: *approximate and solve*

• PPM, • ZEUS, • MUSCL • FFT • multigrid • Gadget cooling • Cloudy cooling • Grackle • Dedner MHD • MHD-CT • Implicit FLD • Moray . . .

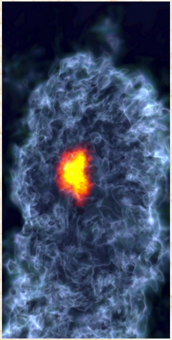
Data Structures: *computer representation*

• Adaptive Mesh Refinement (SAMR) • Eulerian fields • Lagrangian particles . . .

The ENZO MPI-parallel astrophysics application

Although powerful, ENZO's parallel scaling is limited

- HPC advances affect software requirements
  - ENZO was developed starting in 1994
  - Greg Bryan, Michael Norman
  - "massive parallelism" meant $P \approx 10^3$
  - $P \approx 10^7$ today
- Affects different parts of ENZO differently
  - ☺ physics requirements unchanged
  - ☺ numerical methods mostly viable
  - ☹ *data structures* limit ENZO's scalability
- Motivated AMR data structure redesign
  - Enzo-P: "Petascale" branch of ENZO
  - keep ENZO's physics and many methods
  - Cello scalable AMR framework

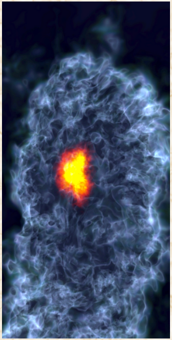


[Sam Skillman, Matt Turk]

The ENZO MPI-parallel astrophysics application

Although powerful, ENZO's parallel scaling is limited

- HPC advances affect software requirements
  - ENZO was developed starting in 1994
  - Greg Bryan, Michael Norman
  - "massive parallelism" meant $P \approx 10^3$
  - $P \approx 10^7$ today
- Affects different parts of ENZO differently
  - ☺ physics requirements unchanged
  - ☺ numerical methods mostly viable
  - ☹ *data structures* limit ENZO's scalability
- Motivated AMR data structure redesign
  - **Enzo-P**: "Petascale" branch of ENZO
  - keep ENZO's physics and many methods
  - **Cello** scalable AMR framework

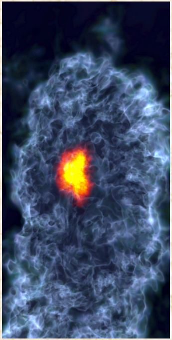


[Sam Skillman, Matt Turk]

The ENZO MPI-parallel astrophysics application

Although powerful, ENZO's parallel scaling is limited

- HPC advances affect software requirements
  - ENZO was developed starting in 1994
  - Greg Bryan, Michael Norman
  - "massive parallelism" meant $P \approx 10^3$
  - $P \approx 10^7$ today
- Affects different parts of ENZO differently
  - ☺ physics requirements unchanged
  - ☺ numerical methods mostly viable
  - ☹ *data structures* limit ENZO's scalability
- Motivated AMR data structure redesign
  - **Enzo-P**: "Petascale" branch of ENZO
  - keep ENZO's physics and many methods
  - **Cello** scalable AMR framework



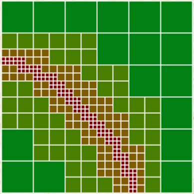
[Sam Skillman, Matt Turk]

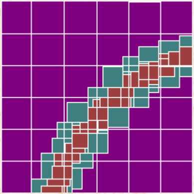
Cello scalable adaptive mesh refinement

Key differences between ENZO and Enzo-P

Enzo-P/Cello

- array of octrees AMR
- Charm++ parallelization
- reusable AMR framework





ENZO

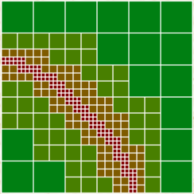
- structured AMR
- MPI+OpenMP parallelization
- non-reusable AMR data structure

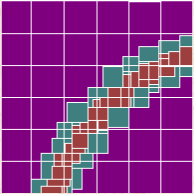
Cello scalable adaptive mesh refinement

Key differences between ENZO and Enzo-P

Enzo-P/Cello

- array of octrees AMR
- Charm++ parallelization
- reusable AMR framework





ENZO

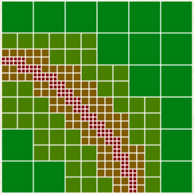
- structured AMR
- MPI+OpenMP parallelization
- non-reusable AMR data structure

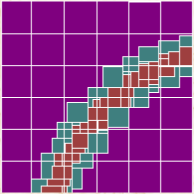
Cello scalable adaptive mesh refinement

Key differences between ENZO and Enzo-P

Enzo-P/Cello

- array of octrees AMR
- Charm++ parallelization
- reusable AMR framework

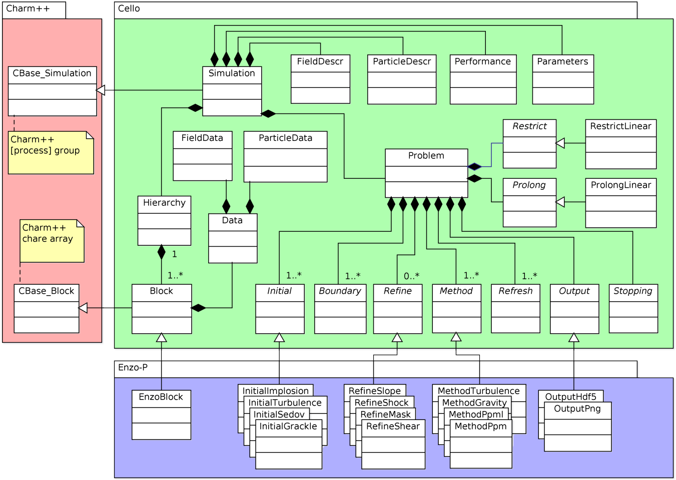




ENZO

- structured AMR
- MPI+OpenMP parallelization
- non-reusable AMR data structure

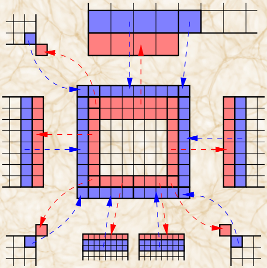
Enzo-P/Cello/Charm++ class organization



Cello distributed adaptive mesh refinement

How field data are communicated between blocks

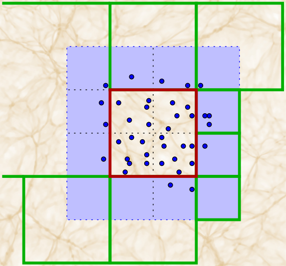
- **Data-driven execution**
  - send `Field` face data when available
  - indexed using bit-coding in hierarchy
  - count face data messages received
  - last receive triggers computation
- **Dynamic task scheduling**
  - multiple `Blocks` per process
  - overlapped communication/computation



Cello distributed adaptive mesh refinement

How particle data are communicated between Blocks

- Communication is required when particles move outside a Block
- This is done using a 4x4x4 array
  - array contains pointers to ParticleData (PD) objects
  - one PD object per neighbor Block

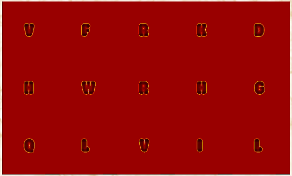


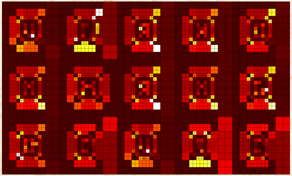
- Migrating particles are
  - `scatter()`-ed to PD array objects
  - sent to associated neighbors
  - `gather()`-ed by neighbors
- One sweep through particles
- One communication step per neighbor

Enzo-P/Cello NSF Blue Waters scaling

"Alphabet Soup" test: hydro and particles

Basic Enzo-P hydrodynamics and particles scalability

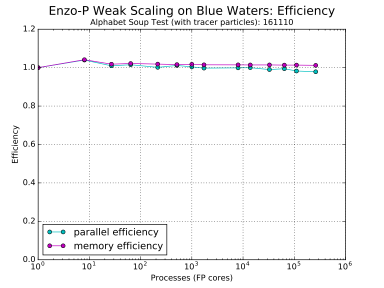


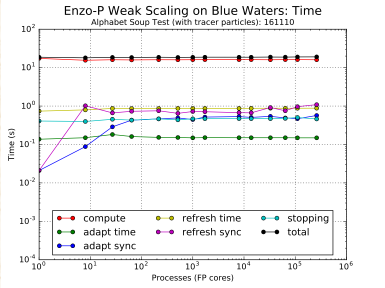


- variation of "array of Sedov Blast" test
- letters instead of spheres
  - inhibits lockstep coarsen/refine
- one blast per BW fp-core
- tested with/without tracer particles
- 32^3 or 24^3 cells per block
- decent sized AMR problem for 2016
  - $262K$ fp-cores
  - $50M$ Blocks
  - $1.7T$ cells; $0.7T$ (cells + particles)
- ENZO would need $72GB$ per process!

Enzo-P/Cello NSF Blue Waters scaling

"Alphabet Soup" test: hydro and particles

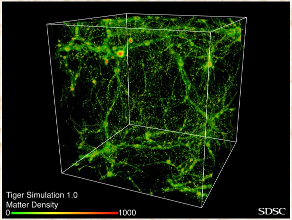




Enzo-P/Cello NSF Blue Waters scaling

"Unigrid Cosmology" test: hydro, particles, gravity

Scaling of basic cosmology simulations

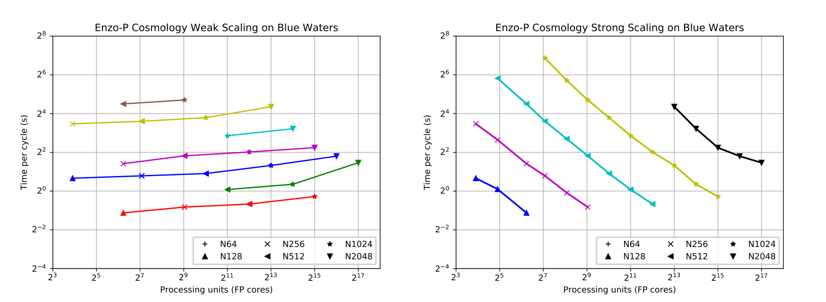


[Renyue Cen]

- PPM hydrodynamics
- "dark matter" particles
- CIC particle-mesh gravity
- multigrid solver—"unigrid" only
- tested up to $131K$ fp-cores

Enzo-P/Cello NSF Blue Waters scaling

“Unigrid Cosmology” test: hydro, particles, gravity



Pursuit of scalable gravity

Linear solvers for $\nabla^2\Phi = 4\pi G\rho$

Recent work has focused on scalable linear solvers

- `EnzoSolverCg`: Conjugate Gradient (CG) Krylov solver
  - Easy to implement (vector updates, matvecs, dot products)
  - May not converge for nonsymmetric problems (AMR)
  - Poor algorithmic scalability if not preconditioned
- `EnzoSolverBiCgStab`: BiCG-STAB Krylov solver
  - Easy to implement
  - Applicable to non-symmetric problems
  - Poor algorithmic scalability if not preconditioned
- `EnzoSolverMg0`: Multigrid V-cycle
  - Trickier to implement (inter-level transfers)
  - Good scalability
  - Limited to non-adaptive meshes (e.g. root-level)

Pursuit of scalable gravity

Linear solvers for $\nabla^2\Phi = 4\pi G\rho$

Recent work has focused on scalable linear solvers

- `EnzoSolverCg`: Conjugate Gradient (CG) Krylov solver
  - Easy to implement (vector updates, matvecs, dot products)
  - May not converge for nonsymmetric problems (AMR)
  - Poor algorithmic scalability if not preconditioned
- `EnzoSolverBiCgStab`: BiCG-STAB Krylov solver
  - Easy to implement
  - Applicable to non-symmetric problems
  - Poor algorithmic scalability if not preconditioned
- `EnzoSolverMg0`: Multigrid V-cycle
  - Trickier to implement (inter-level transfers)
  - Good scalability
  - Limited to non-adaptive meshes (e.g. root-level)

Pursuit of scalable gravity

Linear solvers for $\nabla^2 \Phi = 4\pi G \rho$

Recent work has focused on scalable linear solvers

- `EnzoSolverCg`: Conjugate Gradient (CG) Krylov solver
  - Easy to implement (vector updates, matvecs, dot products)
  - May not converge for nonsymmetric problems (AMR)
  - Poor algorithmic scalability if not preconditioned
- `EnzoSolverBiCgStab`: BiCG-STAB Krylov solver
  - Easy to implement
  - Applicable to non-symmetric problems
  - Poor algorithmic scalability if not preconditioned
- `EnzoSolverMg0`: Multigrid V-cycle
  - Trickier to implement (inter-level transfers)
  - Good scalability
  - Limited to non-adaptive meshes (e.g. root-level)

Pursuit of scalable gravity

Linear solvers for $\nabla^2\Phi = 4\pi G\rho$

Recent work has focused on scalable linear solvers

- `EnzoSolverCg`: Conjugate Gradient (CG) Krylov solver
  - Easy to implement (vector updates, matvecs, dot products)
  - May not converge for nonsymmetric problems (AMR)
  - Poor algorithmic scalability if not preconditioned
- `EnzoSolverBiCgStab`: BiCG-STAB Krylov solver
  - Easy to implement
  - Applicable to non-symmetric problems
  - Poor algorithmic scalability if not preconditioned
- `EnzoSolverMg0`: Multigrid V-cycle
  - Trickier to implement (inter-level transfers)
  - Good scalability
  - Limited to non-adaptive meshes (e.g. root-level)

Enzo-P/Cello Pursuit of scalable linear solvers

Linear solvers for $\nabla^2\Phi = 4\pi G\rho$

Can we build a scalable AMR solver using these solvers?

- "HG" Hierarchical Grid solver (D. Reynolds)
  - `EnzoSolverBiCgStab` solver
  - `EnzoSolverMg0` preconditioner
  - Applicable to AMR problems
  - Better algorithmic scalability
  - Limited parallel scalability (inner-products)
- "DD" Domain Decomposition solver (M.L. Norman)
  - `EnzoSolverMg0` for "long-range" interactions
  - `EnzoSolverBiCgStab` on isolated octrees for "short-range"
  - Final smoothing to clean up subdomain boundaries
  - Promising initial results (memory leak!)
- "AFAC" Asynchronous fast adaptive composite
  - If DD is insufficient, will try multigrid adapted to AMR

Enzo-P/Cello Pursuit of scalable linear solvers

Linear solvers for $\nabla^2\Phi = 4\pi G\rho$

Can we build a scalable AMR solver using these solvers?

- "HG" Hierarchical Grid solver (D. Reynolds)
  - `EnzoSolverBiCgStab` solver
  - `EnzoSolverMg0` preconditioner
  - Applicable to AMR problems
  - Better algorithmic scalability
  - Limited parallel scalability (inner-products)
- "DD" Domain Decomposition solver (M.L. Norman)
  - `EnzoSolverMg0` for "long-range" interactions
  - `EnzoSolverBiCgStab` on isolated octrees for "short-range"
  - Final smoothing to clean up subdomain boundaries
  - Promising initial results (memory leak!)
- "AFAC" Asynchronous fast adaptive composite
  - If DD is insufficient, will try multigrid adapted to AMR

Enzo-P/Cello Pursuit of scalable linear solvers

Linear solvers for $\nabla^2\Phi = 4\pi G\rho$

Can we build a scalable AMR solver using these solvers?

- "HG" Hierarchical Grid solver (D. Reynolds)
  - `EnzoSolverBiCgStab` solver
  - `EnzoSolverMg0` preconditioner
  - Applicable to AMR problems
  - Better algorithmic scalability
  - Limited parallel scalability (inner-products)
- "DD" Domain Decomposition solver (M.L. Norman)
  - `EnzoSolverMg0` for "long-range" interactions
  - `EnzoSolverBiCgStab` on isolated octrees for "short-range"
  - Final smoothing to clean up subdomain boundaries
  - Promising initial results (memory leak!)
- "AFAC" Asynchronous fast adaptive composite
  - If DD is insufficient, will try multigrid adapted to AMR

Enzo-P/Cello Pursuit of scalable linear solvers

Linear solvers for $\nabla^2 \Phi = 4\pi G \rho$

Can we build a scalable AMR solver using these solvers?

- "HG" Hierarchical Grid solver (D. Reynolds)
  - `EnzoSolverBiCgStab` solver
  - `EnzoSolverMg0` preconditioner
  - Applicable to AMR problems
  - Better algorithmic scalability
  - Limited parallel scalability (inner-products)
- "DD" Domain Decomposition solver (M.L. Norman)
  - `EnzoSolverMg0` for "long-range" interactions
  - `EnzoSolverBiCgStab` on isolated octrees for "short-range"
  - Final smoothing to clean up subdomain boundaries
  - Promising initial results (memory leak!)
- "AFAC" Asynchronous fast adaptive composite
  - If DD is insufficient, will try multigrid adapted to AMR

Next steps

- Finish up (finally!) scalable AMR gravity
- Prepare for initial public release
  - Enzo-E developer meeting next week
  - Flux-correction
  - Scalable I/O (Explorations In Exascale I/O library?)
- New physics capabilities
  - MHD solvers
  - Starmaker methods
  - Cosmic ray transport
  - Active particles
- Further performance improvements
  - Block-adaptive time-stepping
  - Accelerator support
  - Dynamic load balancing (SFC?)

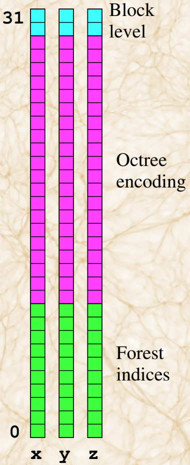
Acknowledgements

Funding for development of Enzo-P/Cello has been provided by the National Science Foundation grants OAC-1835402, SI2-SSE-1440709, PHY-1104819, and AST-0808184.

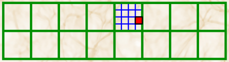
This research is part of the Blue Waters sustained-petascale computing project, which is supported by the National Science Foundation (awards OCI-0725070 and ACI-1238993) and the state of Illinois. Blue Waters is a joint effort of the University of Illinois at Urbana-Champaign and the National Center for Supercomputing Applications.

http://cello-project.org/

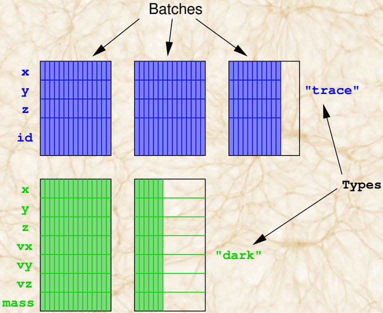
How the Block chare array is indexed



- User-defined chare array indices supported
- Cello indices for Block arrays:
  - 3×10 bits for *array indices*
  - 3×20 bits for the *octree encoding*
  - 6 bits for the *block level*
- Up to 1024^3 array of octrees
- Up to 20 octree levels
- $-31 \leqslant \text{level} \leqslant 31$
- Block id's use index: e.g. B100:11_1:01



How Particle objects store particle data



- multiple particle *types*
- particles allocated in *batches*
  - fixed size arrays
  - fewer new/delete operations
  - efficient insert/delete operations
  - potentially useful for GPU's
- batches store particle *attributes*
  - (position, velocity, mass, etc.)
  - 8,16,32,64-bit integers
  - 32,64,128-bit floats

- particle positions may be floating-point or integers
  - floating-point for storing global positions
  - integers for Block-local coordinates
    - solves reduced precision issue for deep hierarchies
    - less memory required for given accuracy