Future of Enzo

Michael L. Norman
James Bordner
LCA/SDSC/UCSD

SDSC Resources

“Data to Discovery”

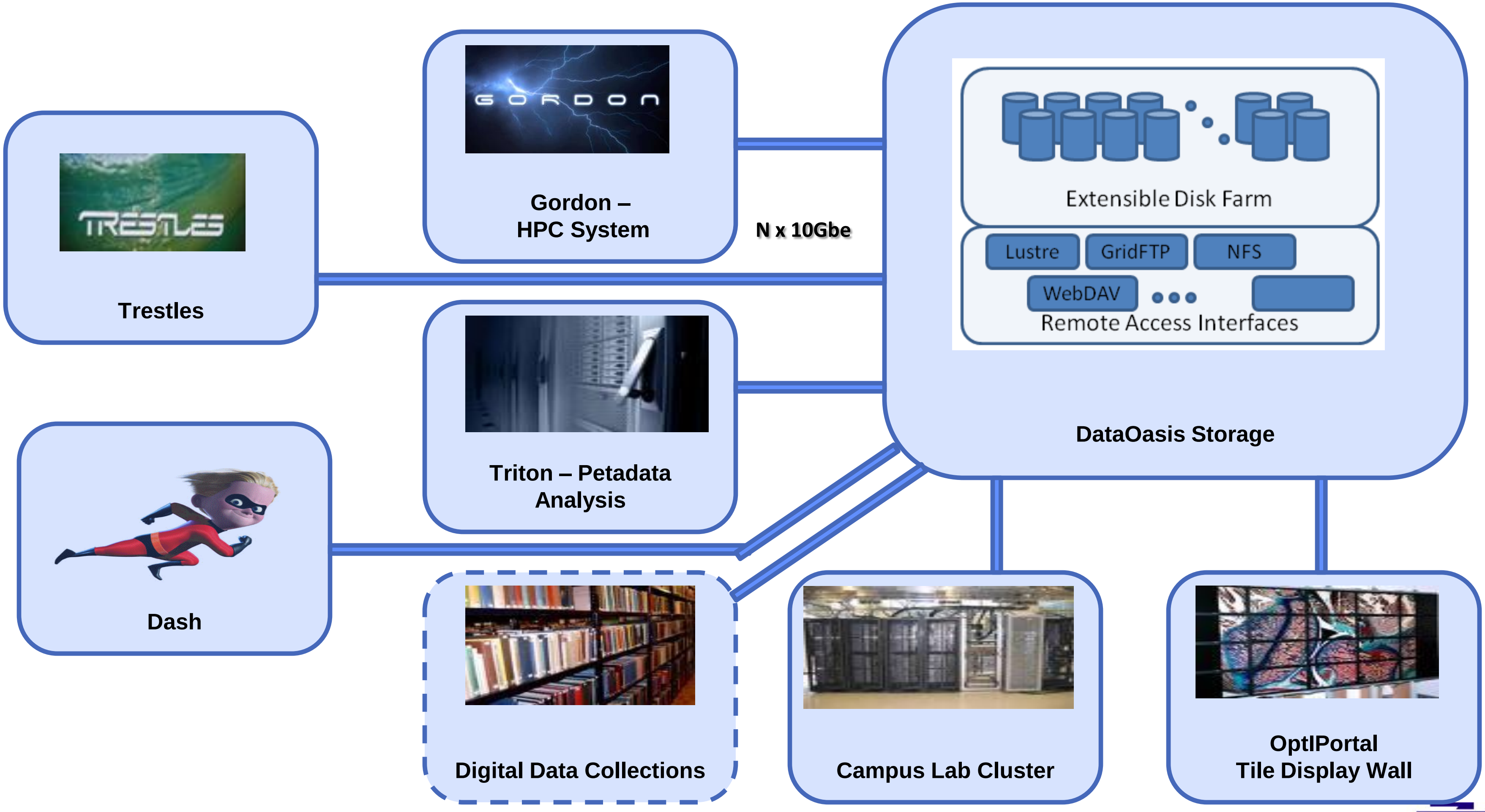
- Host *SDNAP* – San Diego network access point for multiple 10 Gbs WANs
  - ESNet, NSF TeraGrid, CENIC, Internet2, StarTap
- 19,000 Sq-ft, 13 MW green data center
- Host UC-wide co-location facility
  - 225 racks available for your IT gear here
  - can be integrated with SDSC resources
- Host dozens of 24x7x365 “data resources”
  - e.g., Protein Data Bank (PDB) , Red Cross Safe and Well, Encyclopedia of Life,…..

SDSC Resources

- *Data Oasis:* high performance disk storage
  - 0.3 PB (2010), 2 PB (2011), 4 PB (2012), 6 PB (2013)
  - PFS, NFS, disk-based archive
- Up to 3.84 Tbs machine room connectivity
- Various HPC systems

– *Triton* (30 TF)	Aug. 2009	UCSD/UC resource
– *Thresher* (25 TF)	Feb 2010	UCOP pilot
– *Dash* (5 TF)	April 2010	NSF resource
– *Trestles* (100 TF)	Jan 2011	NSF resource
– *Gordon* (260 TF)	Oct 2011	NSF resource

Data Oasis: The Heart of SDSC's Data – Intensive Strategy



Trestles

New NSF TeraGrid resource
in production Jan 1, 2011

Aggregate specs
10,368 cores
100 TF
20 TB RAM
150 TB DISK➔2 PB

Architecture
324 AMD Magny-Cour nodes
32 cores/node
64 GB/node

QDR IB fat tree interconnect

The Era of Data-Intensive Supercomputing Begins

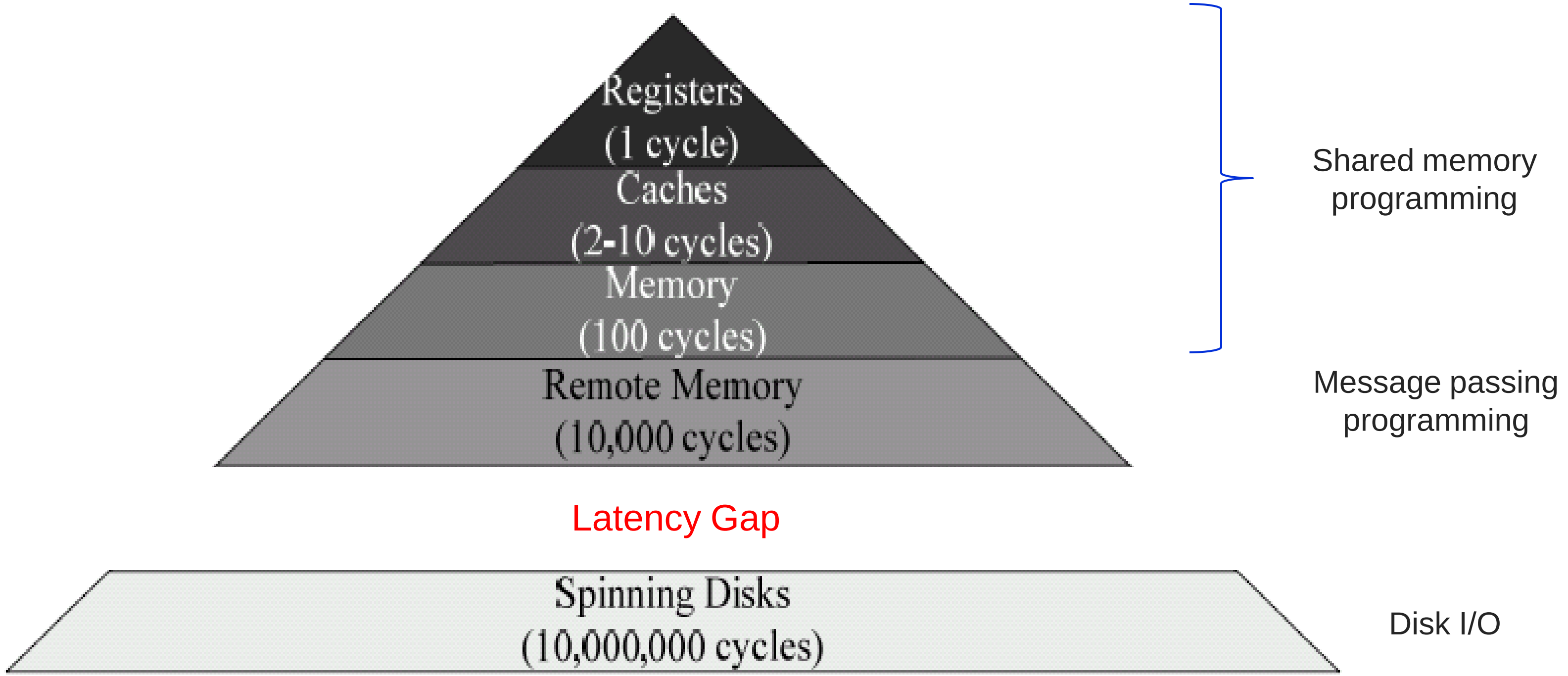
GORDON

Michael L. Norman
Principal Investigator
Interim Director, SDSC

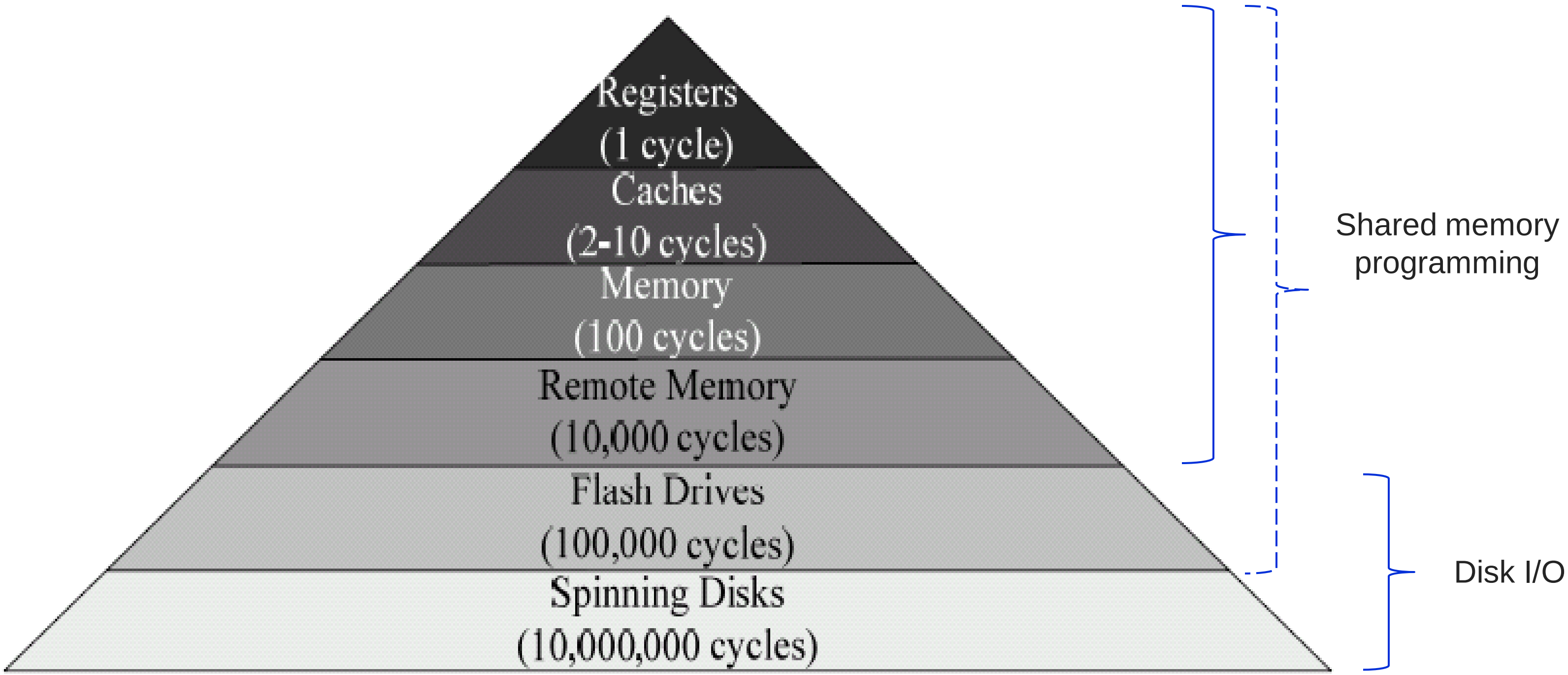
Allan Snavely
Co-Principal Investigator
Project Scientist

COMING SUMMER 2011

The Memory Hierarchy of a Typical HPC Cluster



The Memory Hierarchy of Gordon



Gordon

First Data-Intensive HPC system
In production Fall 2011

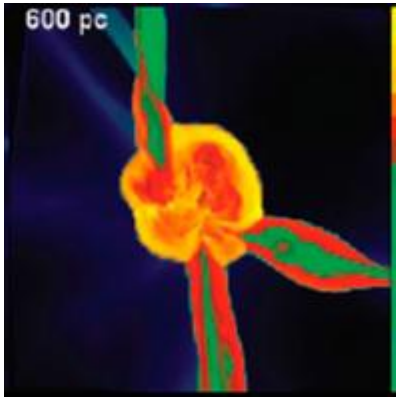
<u>Aggregate specs</u>
16,384 cores
250 TF
64 TB RAM
256 TB SSD (35M IOPS)
4 PB DISK (>100 GB/sec)

<u>Architecture</u>
1024 Intel SandyBridge nodes
16 cores/node
64 GB/node
Virtual shared memory supernodes

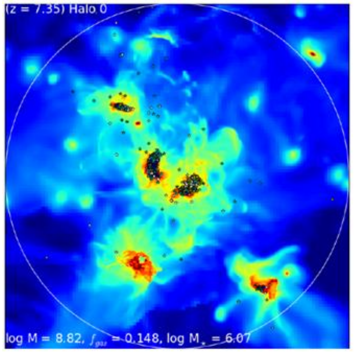
QDR IB 3D torus interconnect


I
Enzo

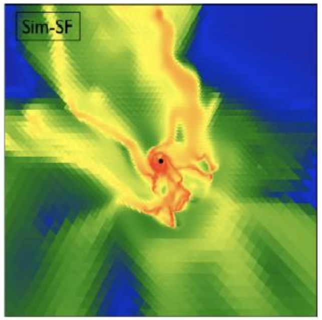
Enzo Science



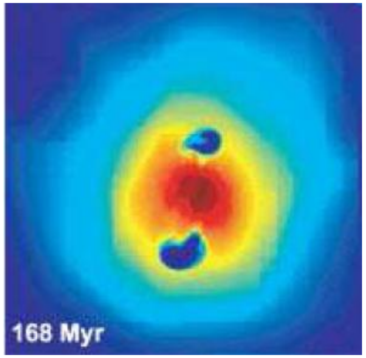
First Stars



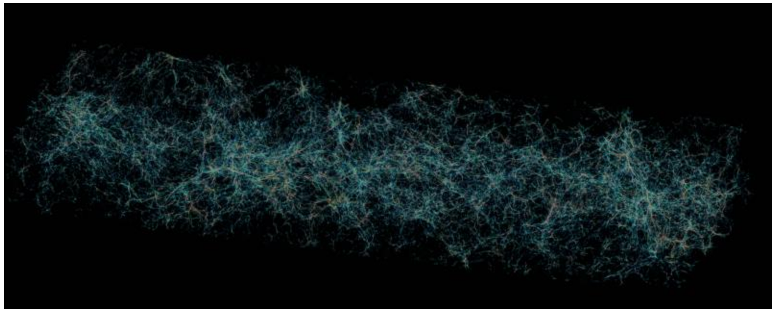
First Galaxies



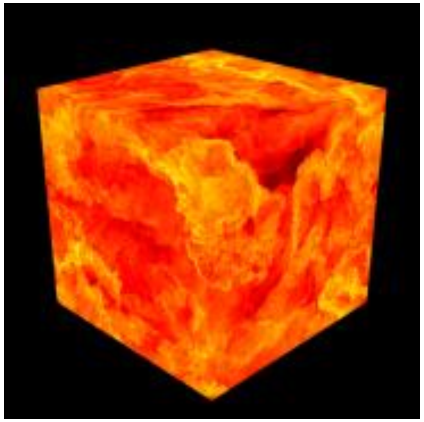
SMBH accretion



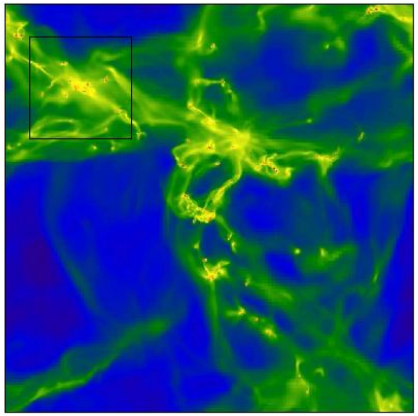
Cluster radio cavities



Lyman alpha forest

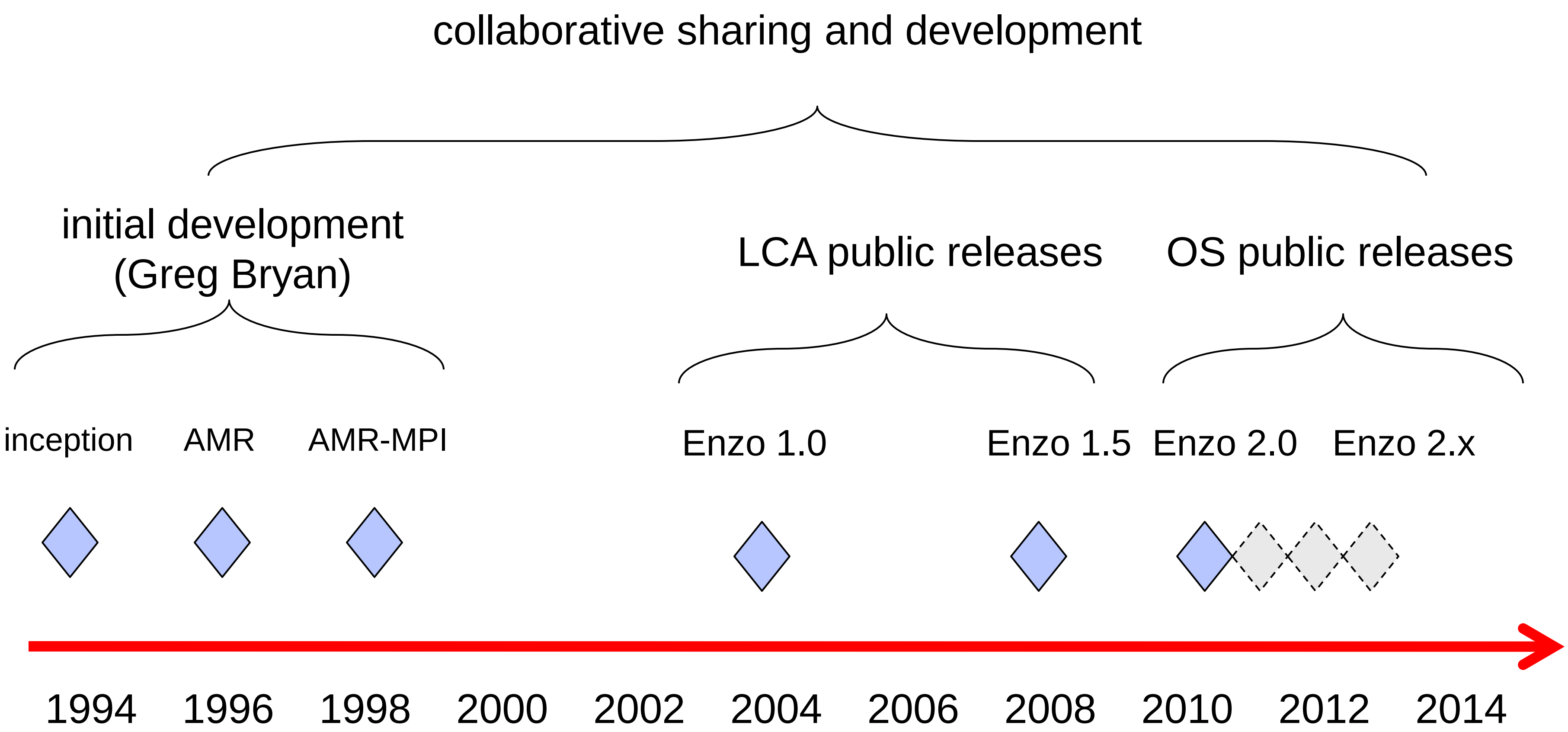


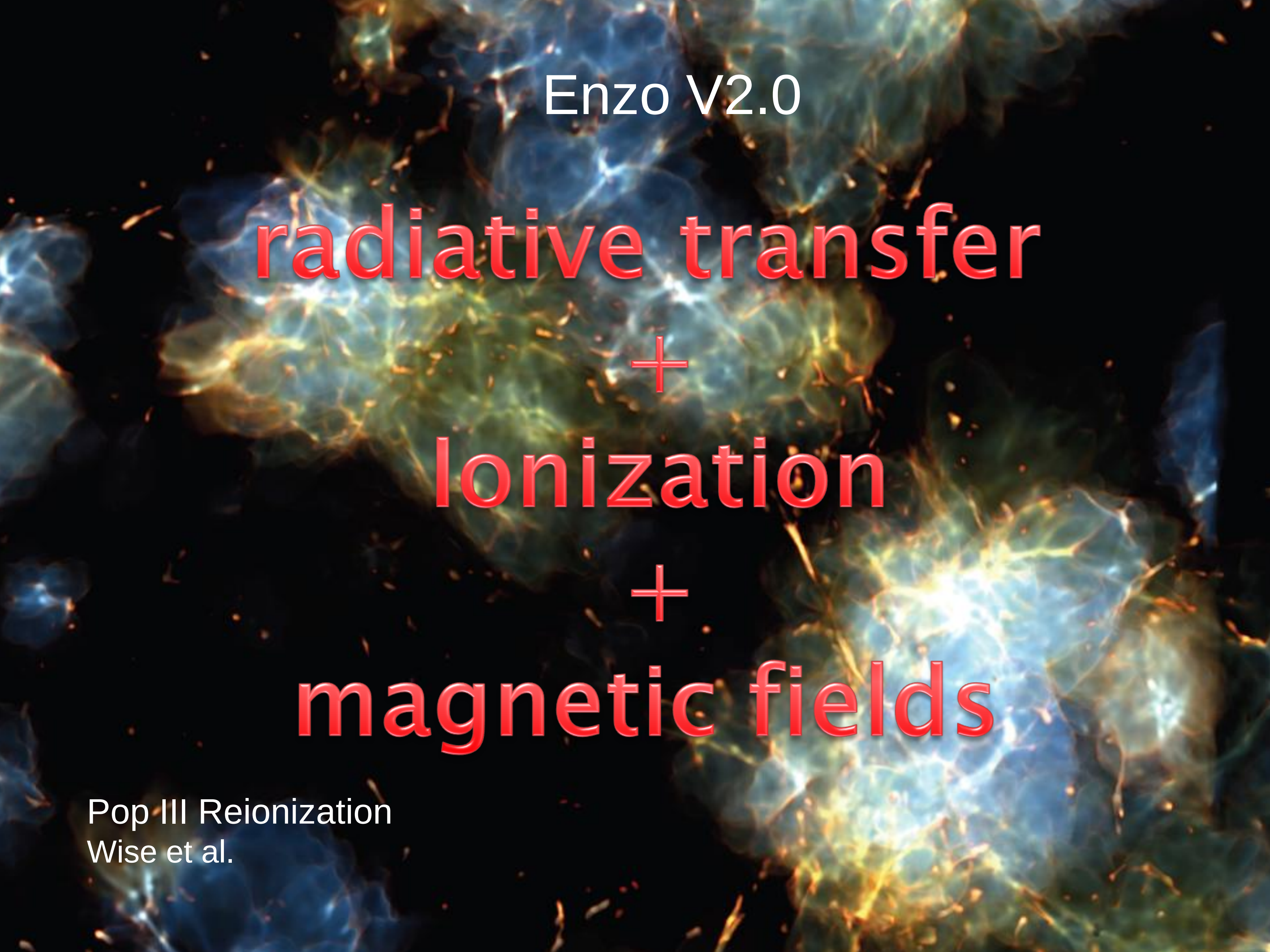
Supersonic turbulence



Star formation

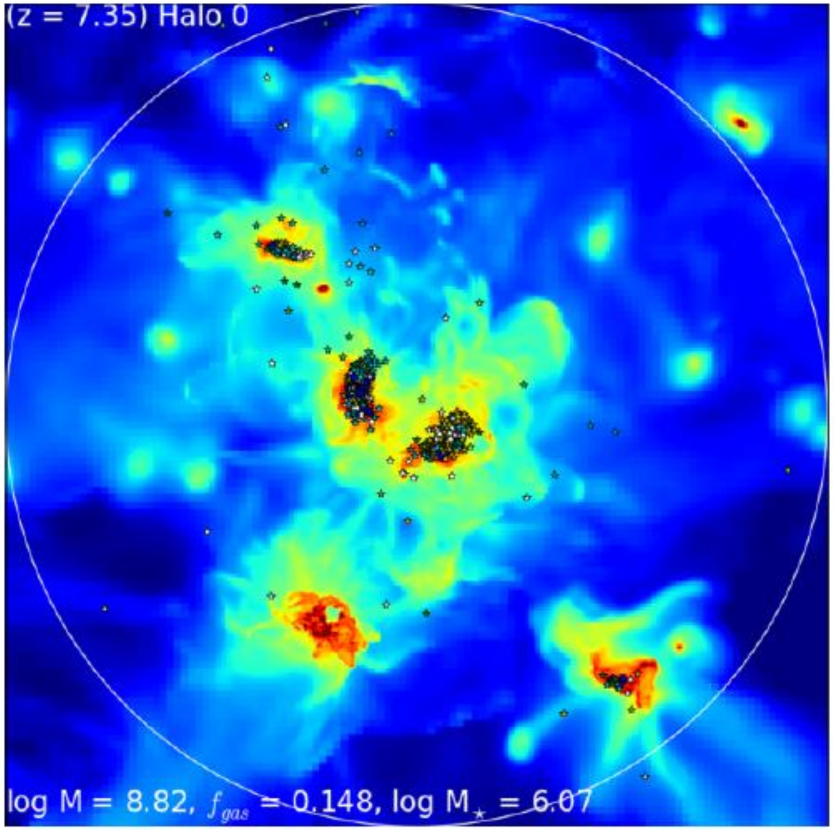
History of Enzo




Enzo V2.0
radiative transfer
+
Ionization
+
magnetic fields
Pop III Reionization
Wise et al.

Current capabilities: AMR vs treecode

First galaxies (ENZO)



Dark matter substructure (PKDGRAV2)

Figure 1: Current capabilities of cosmological simulations. **Left**: EnzoAMR simulation of a primeval galaxy at z=7.35. From [86] **Right**: PKDGRAV2 simulation of dark matter substructure of a Milky Way size halo at z=0. From [66].

Enzo-P

ENZO: THE NEXT GENERATION

Enzo-P

ENZO:THE NEXT GENERATION

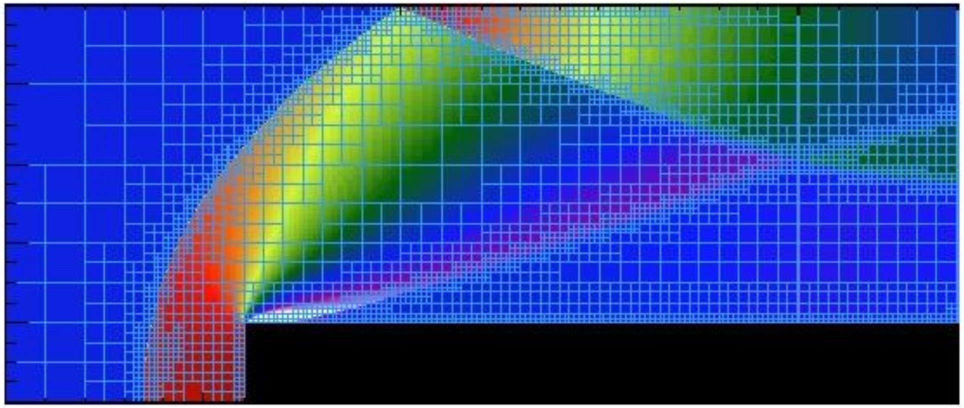
- ENZO's AMR infrastructure limits scalability to $O(10^4)$ cores
- We are developing a new, extremely scalable AMR infrastructure called *Cello*
  - http://lca.ucsd.edu/projects/cello
- ENZO-P will be implemented on top of Cello to scale to $10^{6\text{-}8}$ cores

The Cello Project

Enzo: The Next Generation

- Core ideas
  - Take the best fast N-body data structure (hashed KD-tree) and “condition” it for higher order-accurate fluid solvers
  - Flexible, dynamic mapping of hierarchical tree data structure to the hierarchical parallel architecture
    - Object oriented design
  - Build on best available parallel middleware for fault-tolerant, dynamically scheduled concurrent objects (Charm++)
  - Easy ports to MPI, UPC, OpenMP, …..

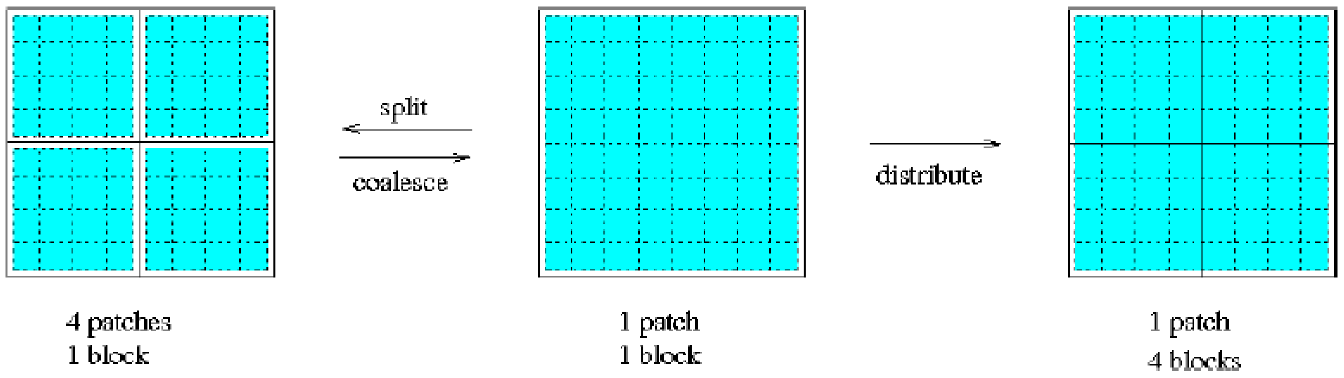
Cello AMR approach



- Based on octrees rather than SAMR for scalability
  - octree AMR has scaled to > 200K cores
  - mesh data associated with leaf nodes only
- Enhancements to address other issues
  - **patch coalescing** to reduce AMR overhead
  - **targeted refinement** for deep AMR problems

Cello AMR

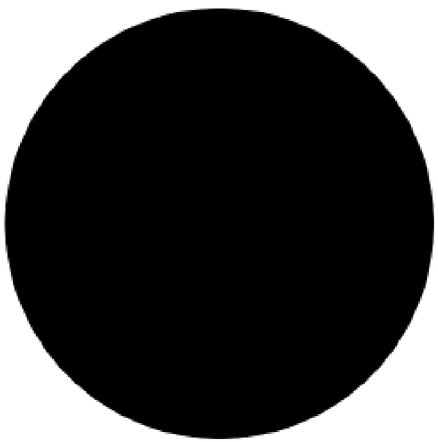
Enhancement 1: Patch coalescing



- Coalesce patches into larger one when possible
- Split a patch into smaller ones when necessary
- Maintain task size control using "blocks"

Cello AMR

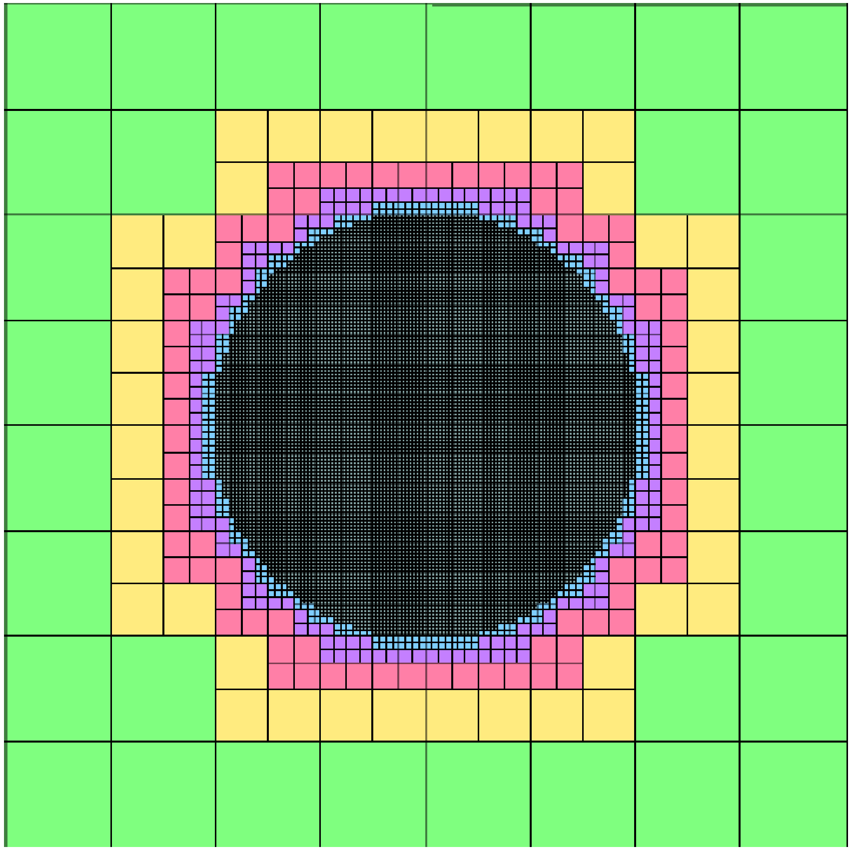
Enhancement 1: Patch coalescing



- Assume you want to refine on a circle
- quadtree refinement has 18517 patches
- coalescing patches reduces to 789 patches

Cello AMR

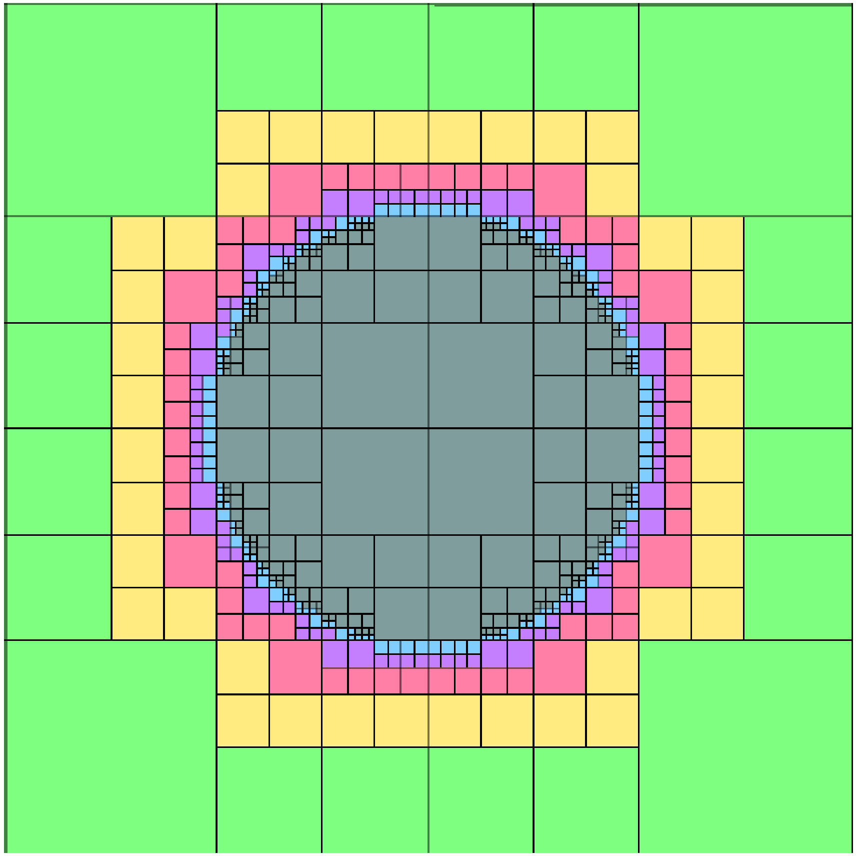
Enhancement 1: Patch coalescing



- Assume you want to refine on a circle
- quadtree refinement has 18517 patches
- coalescing patches reduces to 789 patches

Cello AMR

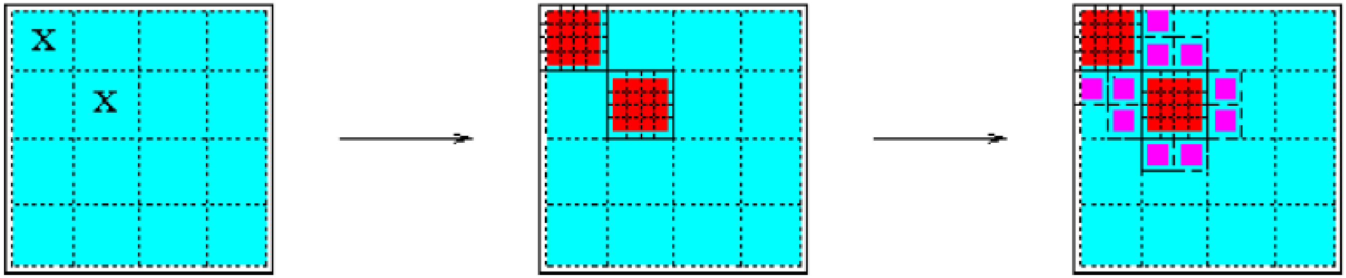
Enhancement 1: Patch coalescing



- Assume you want to refine on a circle
- quadtree refinement has 18517 patches
- coalescing patches reduces to 789 patches

Cello AMR

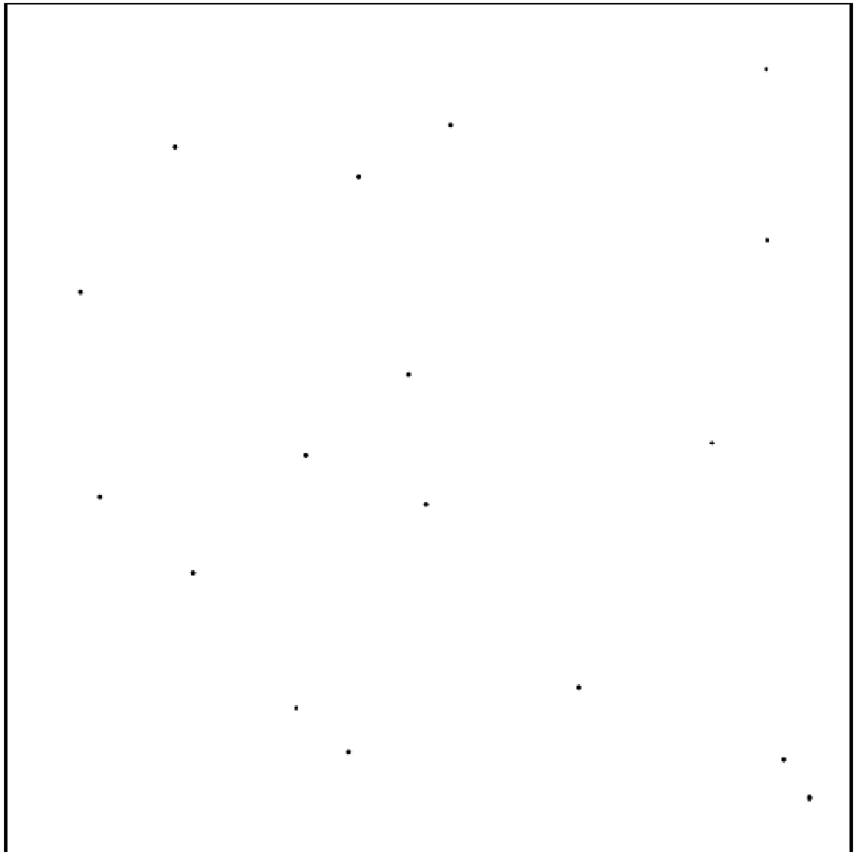
Enhancement 2: Targeted refinement



- Refine by $r = 4$ instead of $r = 2$
- Refinement is more localized
- Can restore $r = 2$ jumps by "backfilling" levels
- Backfill patch locations known implicitly—nominal storage

Cello AMR

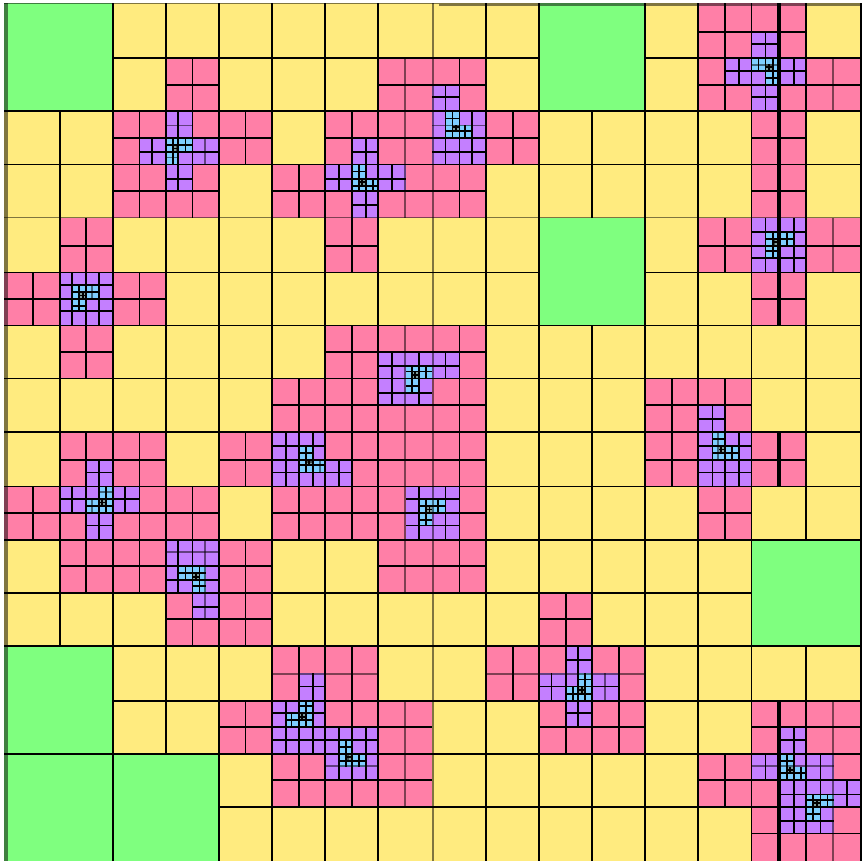
Enhancement 2: Targeted refinement



- Assume you want to refine on point sources
- quadtree refinement with $r = 2$ has 2137 patches
- targeted refinement with $r = 4$ has 158 patches

Cello AMR

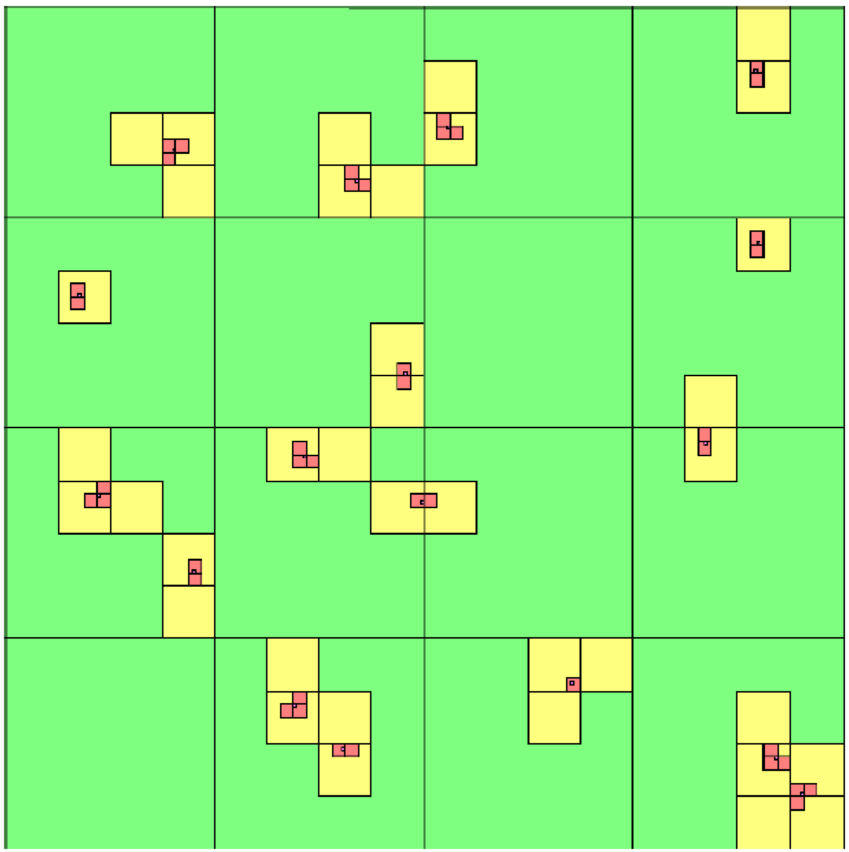
Enhancement 2: Targeted refinement



- Assume you want to refine on point sources
- quadtree refinement with $r = 2$ has 2137 patches
- targeted refinement with $r = 4$ has 158 patches

Cello AMR

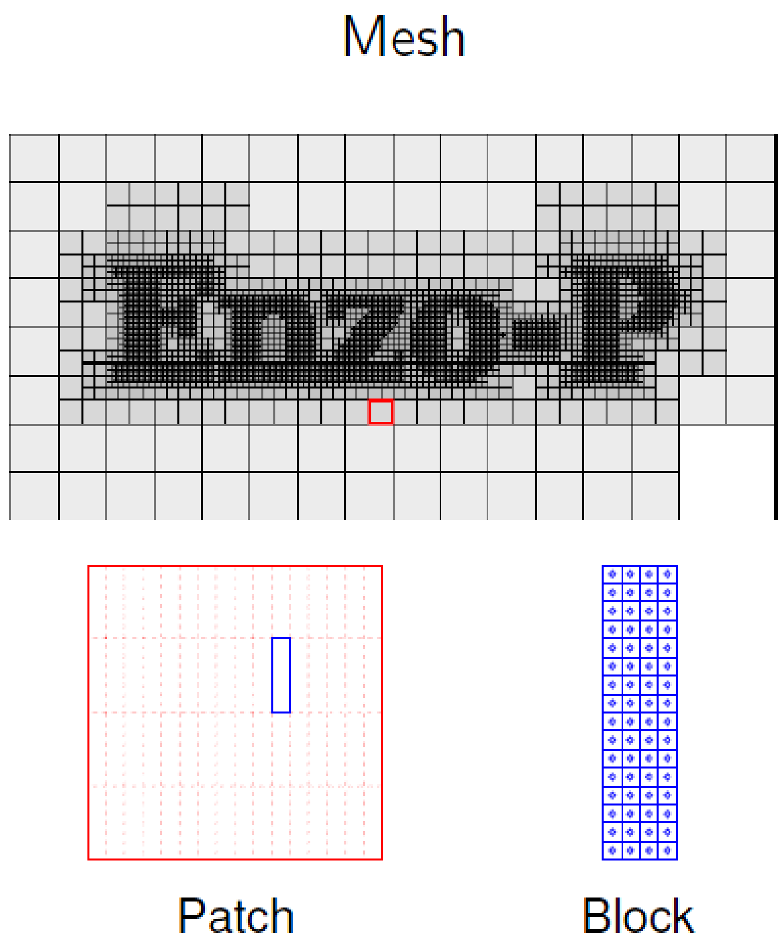
Enhancement 2: Targeted refinement



- Assume you want to refine on point sources
- quadtree refinement with $r = 2$ has 2137 patches
- targeted refinement with $r = 4$ has 158 patches

Cello Mesh data structure

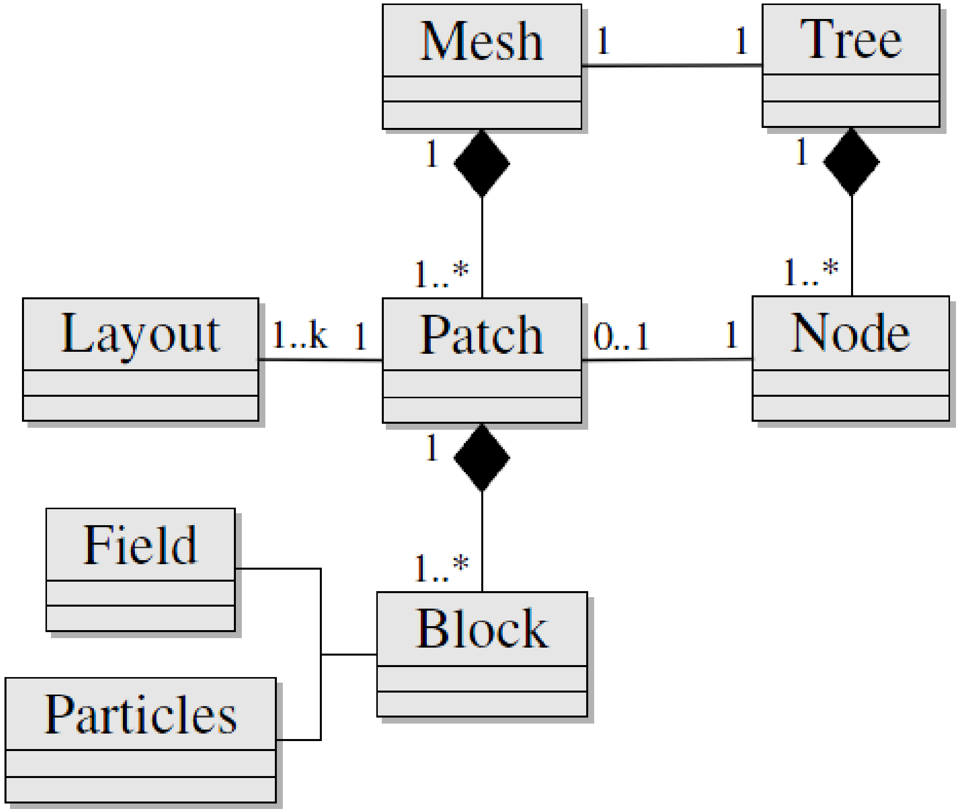
Core `Mesh` classes



- Decouple mesh refinement from data distribution
  - in Enzo, `grid` = parallel task
  - in Cello, `Patch` ⊇ `Block` = parallel task
  - `Block` size can be optimized independently of `Patch` size
    - target specialized computational kernels
    - increased parallelism
    - improved load balancing
    - reduced memory fragmentation
- "Unigrid" when possible; AMR when necessary
  - leverage unigrid performance and scalability
  - `Patches` encapsulate parallel unigrid subproblems
  - $O(1)$ metadata for full unigrid problem ($O(P)$ for Enzo)

Cello Mesh data structure

Mesh related classes



Cello Mesh data structure

Mesh related classes

- Mesh: full AMR hierarchy
- Patch: region of uniform resolution
  - Cello unigrid problem degenerates to single Patch
- Block: basic distributed data unit / parallel task
  - MPI: e.g. one Block per process in Cartesian topology
  - CHARM++: one Block per 3D "chare array"
  - GPU / OMP / UPC support planned
- Layout: specifies how to distribute Blocks in a Patch
  - Block size, process range, neighbor pointers, etc.
  - hierarchical parallelism through multiple Layouts
- Block data: Field, Particles, etc.
- Tree, Node: bare-bones octree data structure
  - Nodes are only objects replicated across machine
    - small nodes: ≤ 24 bytes (> 1500 bytes/grid for Enzo)
    - fewer nodes: e.g. 1 instead of P for unigrid case

Cello Status

- Software design completed
  - 200 pages of design documents
- ~20,000 lines of code implemented
- PPM hydro code for uniform grid with Charm++ parallel objects initial prototype
- Next up: AMR
- Seeking funding and potential users