EXPERIENCES WITH AMR CO-DESIGN FROM THE PERSPECTIVE OF AN APPLICATION USING AMR

JULY 10, 2017
ANSHU DUBEY
MATHEMATICS AND COMPUTER SCIENCE DIVISION
ARGONNE NATIONAL LABORATORY

COLLABORATORS – DAN KASEN, BRONSON MESSER, ANN ALMGREN, MIKE ZINGALE, RAPH HIX, KLAUS WEIDE

Core Questions in Nuclear Astrophysics

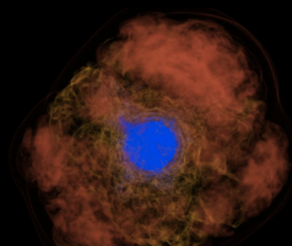
- How were the elements in the universe made (r-process nucleosynthesis)?
- What is the behavior of matter at extreme densities (nuclear EOS)?
- What are the fundamental properties of neutrinos?

Stellar explosions are laboratories for studying nuclear physics in regimes that are inaccessible in terrestrial experiments

Challenge problems to simulate

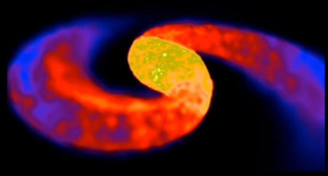
core-collapse supernovae

explosion of massive stars reaching extremes of nuclear burning and neutrino interactions

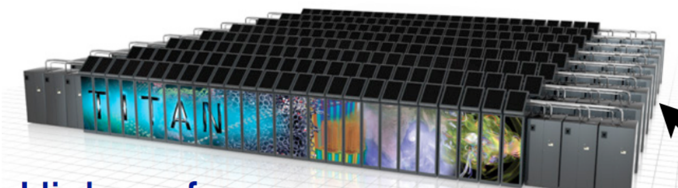


neutron star mergers

inspiral and coalescence of compact stars reaching extremes of gravity and density



Exascale simulations provide the link connecting observations on astrophysical scales to nuclear physics on microscopic scales, which is needed to guide and interpret both astro and nuclear experiment.



High performance computing

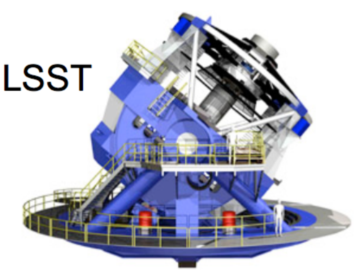
Origin of the elements
Matter at extreme density
Neutrino physics

Simulation of stellar explosions

Astro experiment

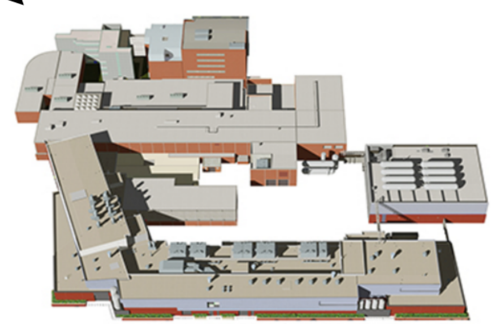
optical telescopes (ZTF, DES, LSST)
x-ray, gamma-ray satellites (nuStar)
neutrino detectors (DUNE)
gravitational wave exp. (LIGO)

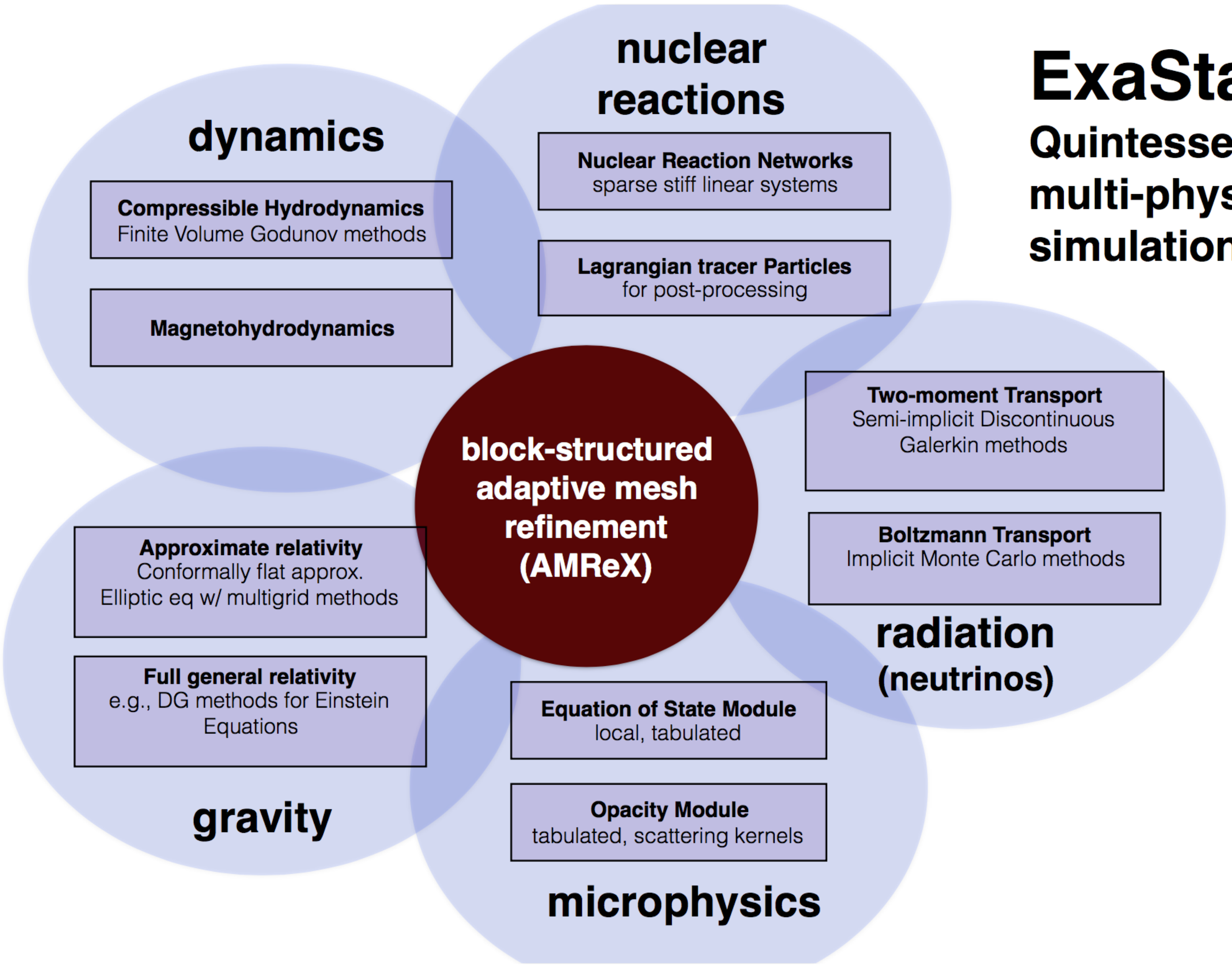
LSST



Nuclear experiment

e.g., Facility for Rare Isotope Beams





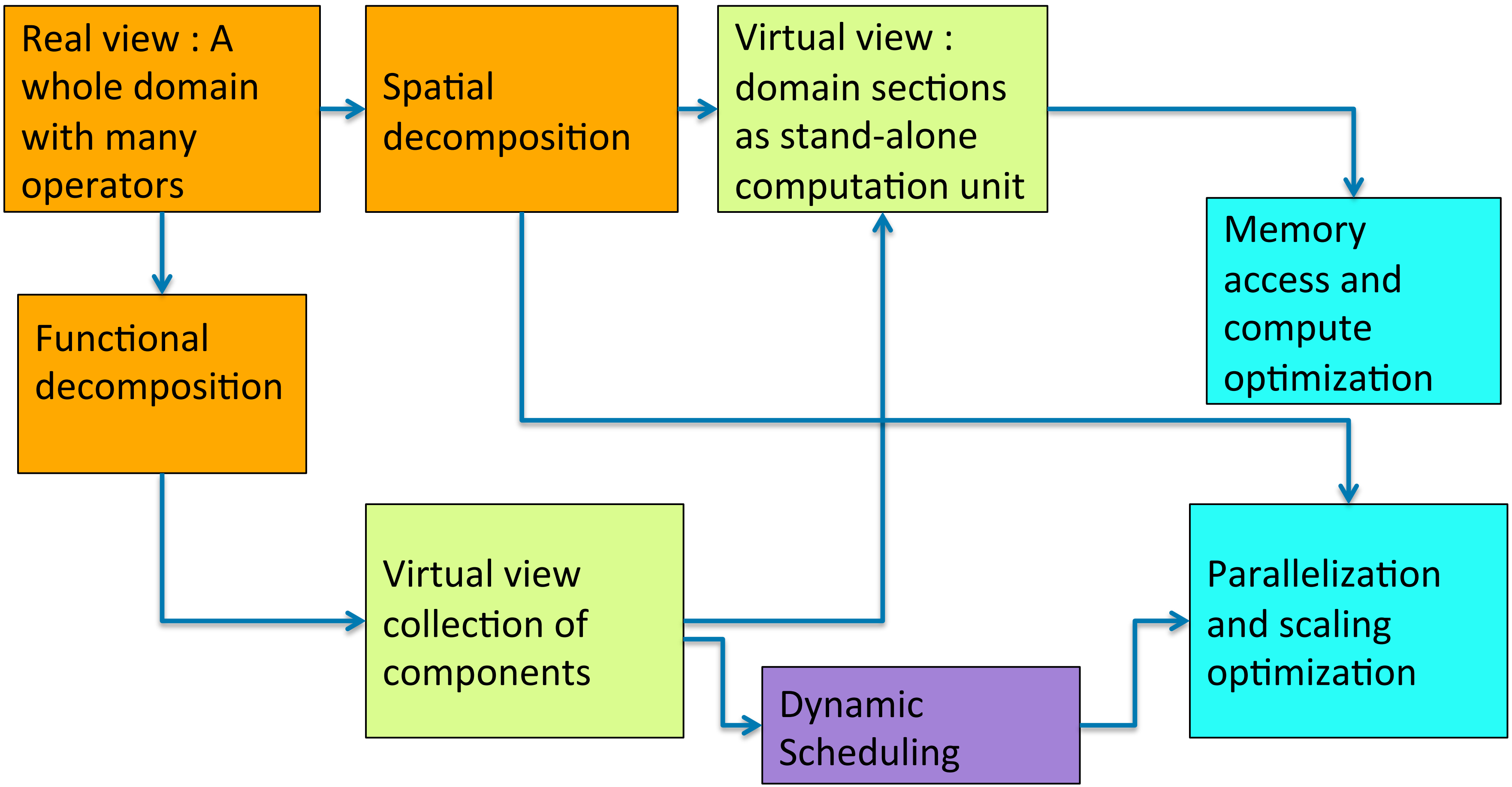
ExaStar

Quintessential multi-physics simulation

TAMING THE COMPLEXITY

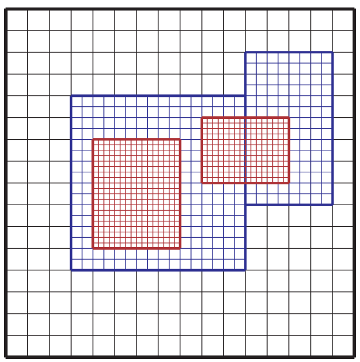
- Logically separable functional units of computation
  - Infrastructure
  - Solvers
  - Monitors
- Encode the logical separation (modularity) into a framework
  - Infrastructure units being the backbone
- Define interfaces through which the modules can interact with each other
- Establish separation of concerns

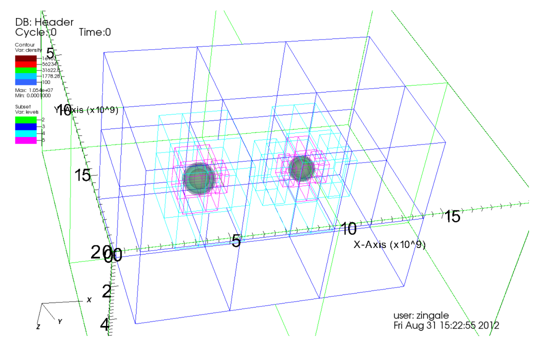
SEPARATION OF CONCERNS



BLOCK-STRUCTURED AMR DEFINES THE DATA LAYOUT

In block-structured AMR, the solution is defined on a hierarchy of levels of resolution, each of which is composed of a union of logically rectangular grids/patches





- Patches change dynamically
- More generally, patches may not be fixed size and may not have unique parent

Data is in the form of

- mesh data
- Particles

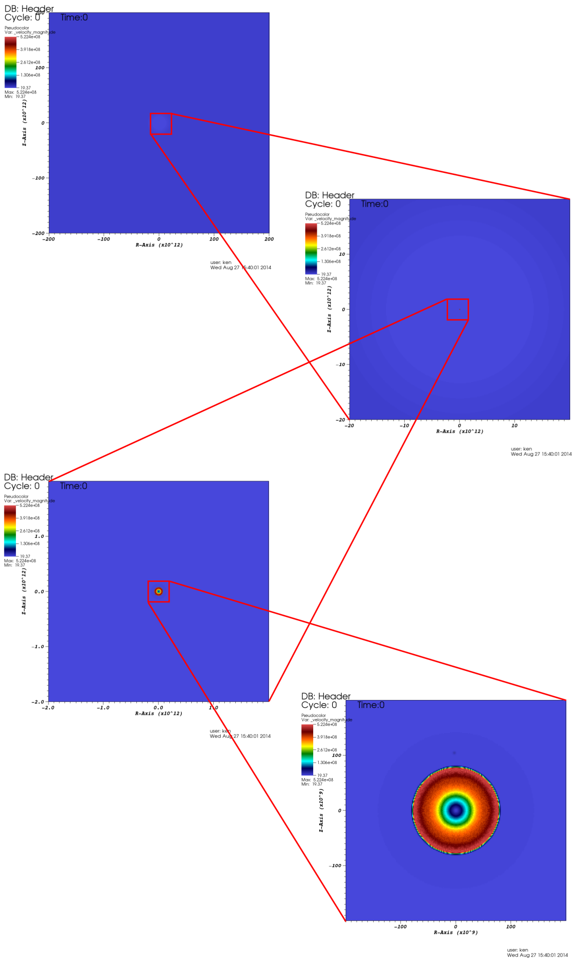
AMR DOES NOT DEFINE THE DISCRETIZATIONS

- ❑ Block-structured AMR does not define the algorithm or the spatial or temporal discretizations

- ❑ Time-stepping options including
  - ❑ Advancing all levels with a single time step
  - ❑ Subcycling in time (finer levels take multiple time steps for each coarser time step)
  - ❑ Optimal subcycling (subcycle between some but not all levels as determined by the time step constraints)

Power of 10 (CASTRO)

Each close-up is by a factor of ten

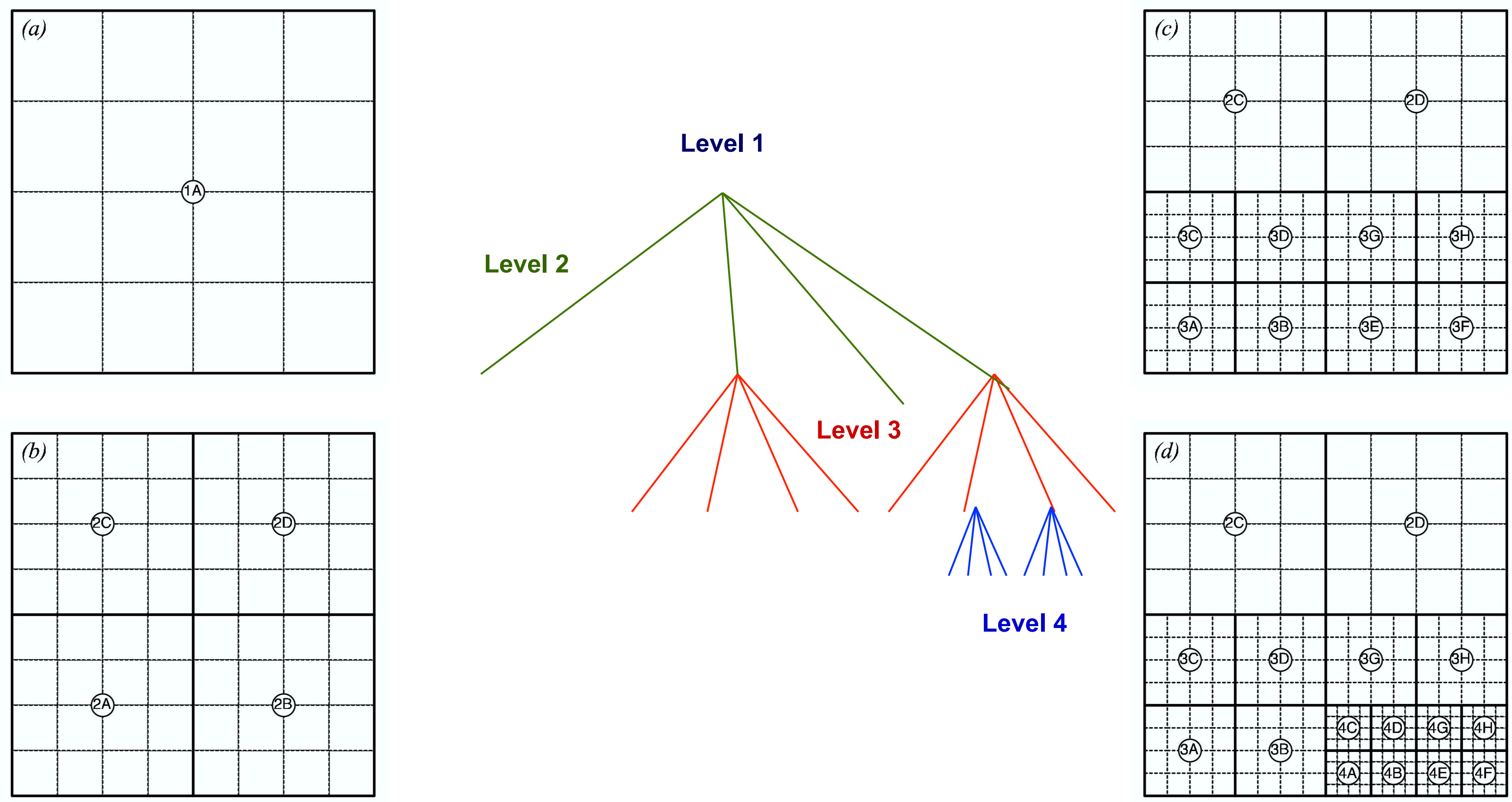


AMR PROVIDES NATURAL OPPORTUNITIES FOR PARALLELISM

- AMR provides a natural framework for reducing the memory footprint and computational cost of a structured grid simulation

- The infrastructure to support block-structured AMR naturally supports hierarchical parallelism:
  - Coarse-grained dynamic load balancing due to decomposition into multiple grids at multiple levels
  - Fine-grained optimization opportunities due to regular patches of data

OCT-TREE AMR 2D BLOCK MAP



- ❑All blocks have same dimensions
- ❑Blocks have parent-child relation

FROM APPLICATION END

- Letting go of mesh awareness
  - Change from pull to push model
  - Earlier sequence of actions
    - Get list of blocks and loop over them
    - Get meta information based on blockid
    - Apply operator
  - New way
    - Let the iterator handle ready block
    - Meta information encoded with the block
    - Apply operator

Before -----

```
 do dr_nstep = dr_nbegin, dr_nend

    call Grid_getLocalNumBlks(localNumBlocks)
    call Grid_getListOfBlocks(LEAF,blockList,blockCount)
    call Hydro( blockCount, blockList, &
             dr_simTime, dr_dt, dr_dtOld)
end do
```

In physics

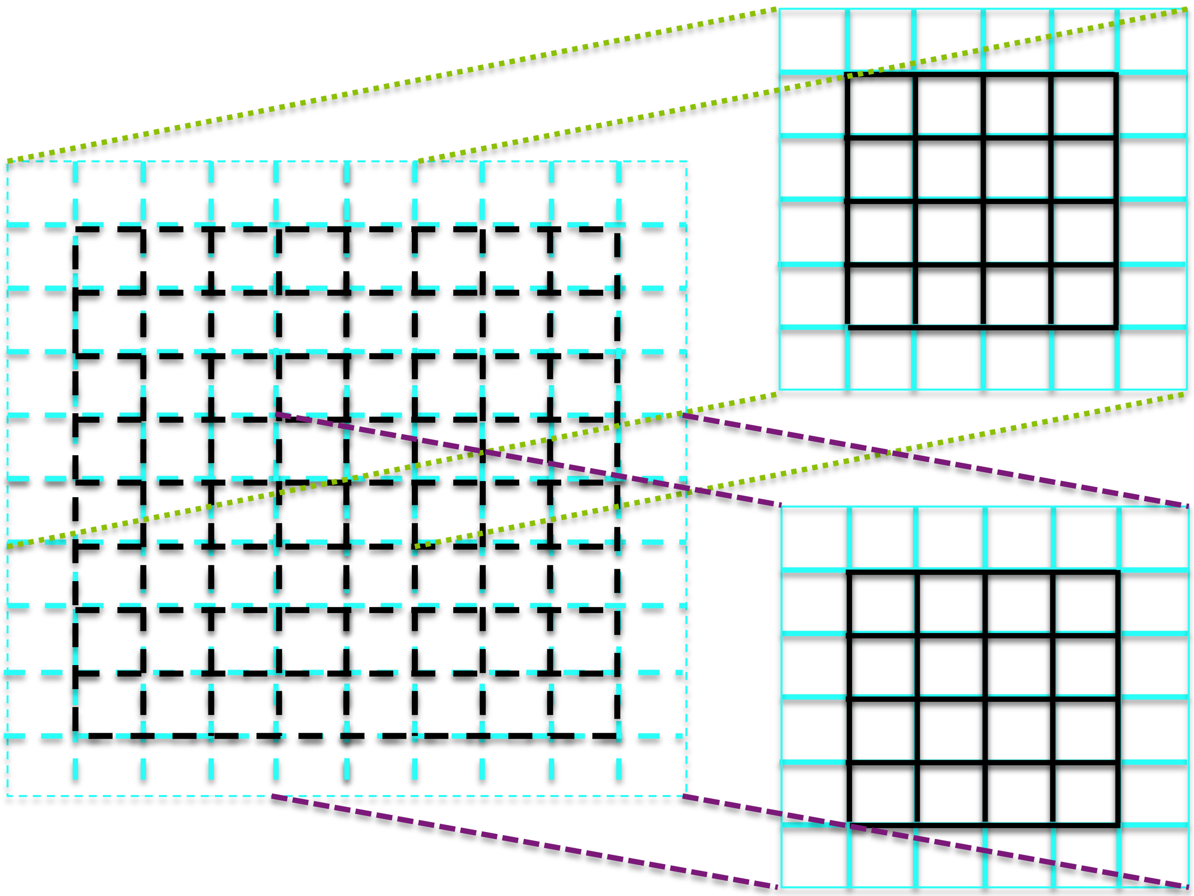
```
 call Grid_fillGuardCells(CENTER,ALLDIR)
  do i=1,blockCount             !LOOP 1
     blkid = blockList(i)
     call Grid_getDeltas(blkid,…)
     call Grid_getBlkIndexLimits(blkid…)
     call Grid_getBlkPtr(blkid….)
     call hy_block
```

Now -----

```
 do dr_nstep = dr_nBegin, dr_nend
    call Grid_fillGuardCells(CENTER,ALLDIR)
    do level=1,maxLev
       call famrex_multivab_build()
      call famrex_mviter_build(mvi, phi(level))
      do while(mvi%next())
        bx = mvi%tilebox() !! Extract other meta info
       Uout => phi(level)%dataptr(mvi)
     call Hydro(Limits,Uout,dr_simTime, dr_dt, dr_dtOld)
   end do
```

LOGICAL TILING

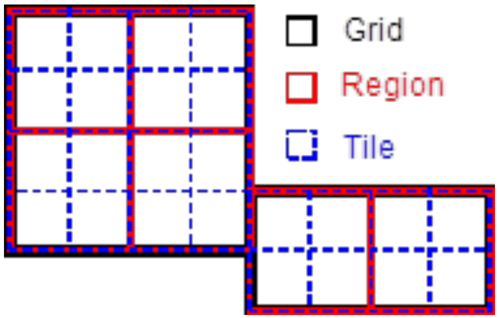
- The solver view of patches
  - Not fine-grained enough by itself
  - Too small and auxiliary memory needs dominate over real memory



- Eliminate in-place updates
- The view from the source data
- The ghost cells aren't really duplicated
- More than one block reads the same cell
- For one it is the active cell, for all others it is the ghost cell

LOGICAL TILING CAN REDUCE COST ON A SINGLE CORE

- With logical tiling, the data layout is unchanged but the unit of work is a tile rather than a grid
- Can hide tiling in the iterator so is invisible to the application
- Leads to more efficient memory access



1 core of Edison 128^3 domain

	GNU compiler		Intel complier	
Tile Size	Time(s)	Speedup	Time(s)	Speedup
128 × 4 × 4	8.5	3.4	8.7	1.8
128 × 8 × 8	9.0	3.2	9.6	1.6
128 × 16 × 16	9.6	3.0	10.5	1.5
128 × 32 × 32	23.7	1.2	10.4	1.5
128 × 64 × 64	24.4	1.2	10.9	1.4
no tiling	28.6	–	15.5	–

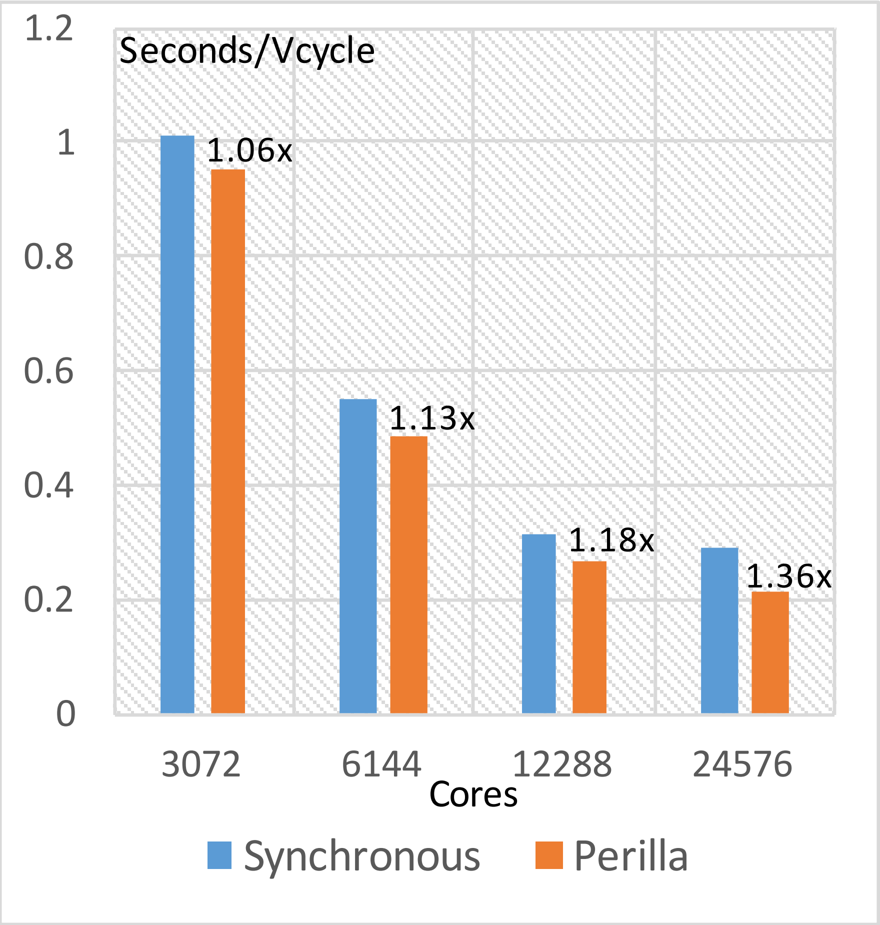
Courtesy of Weiqun Zhang, Didem Unat and Tan Nguyen

ASYNCHRONOUS EXECUTION

- Barriers are the easy way to reconcile dependencies
  - Take away the option of pipelining and/or overlapping
- With hierarchical spatial and functional decomposition rich collection of tasks
  - Articulate dependencies explicitly
  - Let the framework find the unit of computation that is ready and hand it to client code with all the necessary data
    - Under the hood, framework can be managing dependencies
    - If client code assumes not-in-place update each of the tiles is a task with neighborhood dependencies
- Can be made into build or run environment specifications through appropriate parameterization

AMR METADATA CAN FACILITATE ASYNCHRONOUS RUNTIME

- At the lowest level, we can use an asynchronous runtime
  - Leverages the metadata already created to simplify the process of constructing a task graph
  - Hides communication overhead with asynchronous messages
  - NUMA-aware: communication within a compute node is fast



- Results show up to 1.36x speedup for 2K^3 geometric multigrid solver on 24K cores on Edison

Courtesy of Tan Nguyen, Didem Unat, Weiqun Zhang

PUTTING IT ALL TOGETHER

- The construction of operators
  - Express computation at a higher level with abstraction
  - Specify the part of the domain, and the conditions under which the operators apply
- Mix-mode parallelism
  - Parameters to control the degree of tiling or other forms of mix-mode parallelism
    - Could be handed to the compiler when technology arrives
  - Framework forms the data containers
- Dynamic tasking
  - Smarter iterators that are aware of mix-mode parallelism and dependencies
  - The iterating loops give up control and do while loops

-CO-DESIGN IS ENABLING BETTER SEPARATION OF CONCERNS BETWEEN INFRASTRUCTURE AND PHYSICS

-COMPLEX ORCHESTRATION OF PARALLELISM OFFLOADED TO AMREX

-NON INVASIVE BUT PERVASIVE CHANGES TO THE OPERATORS