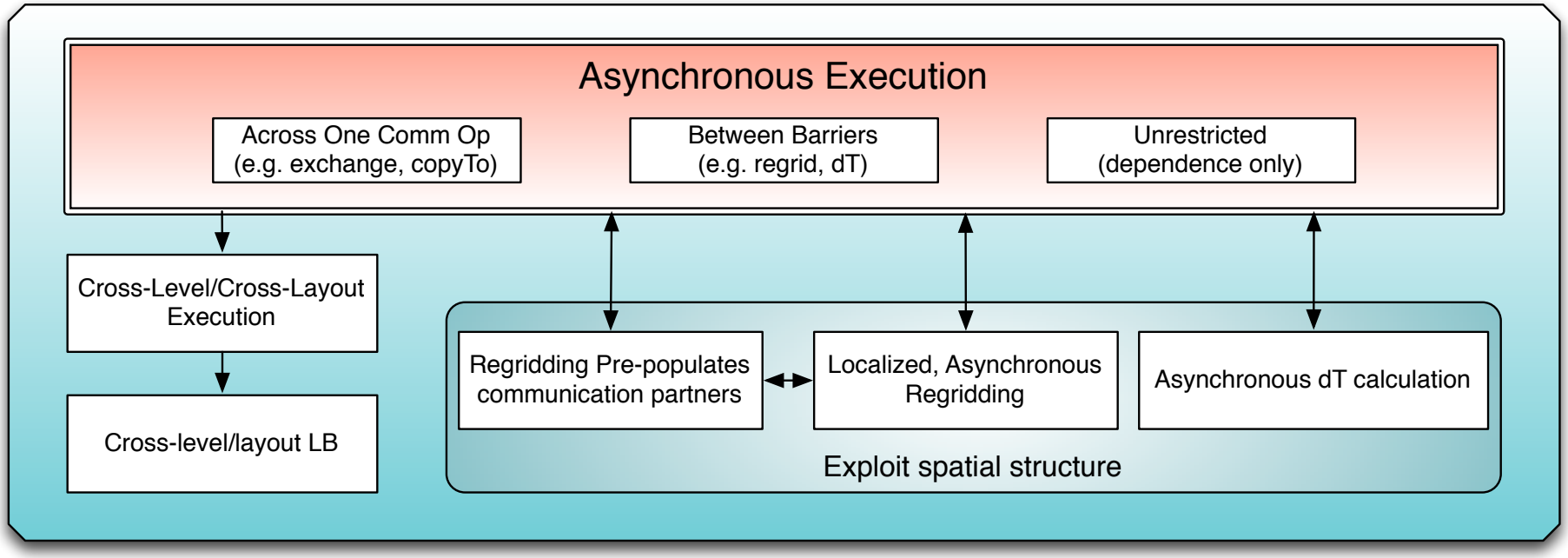
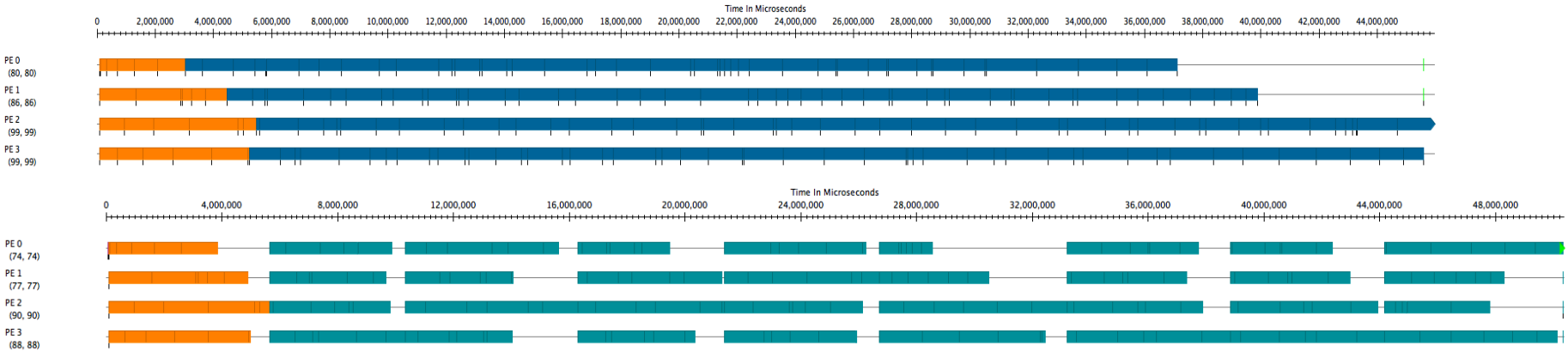
Fully Asynchronous Execution of Adaptive Mesh Refinement Applications

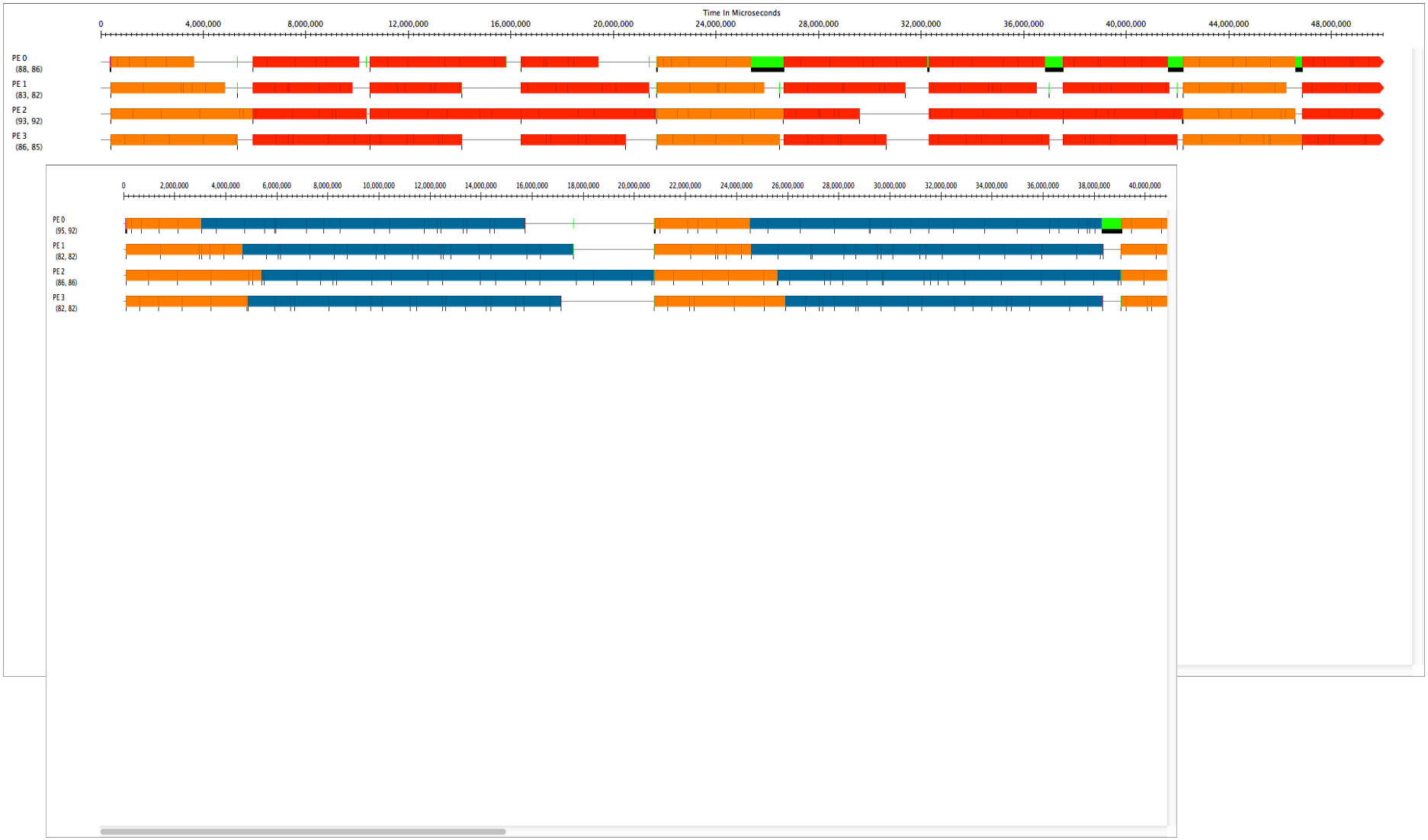
Phil Miller



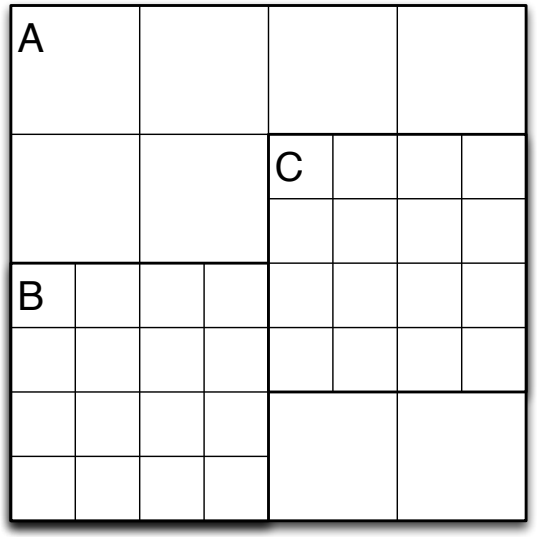
Why synchronization hurts

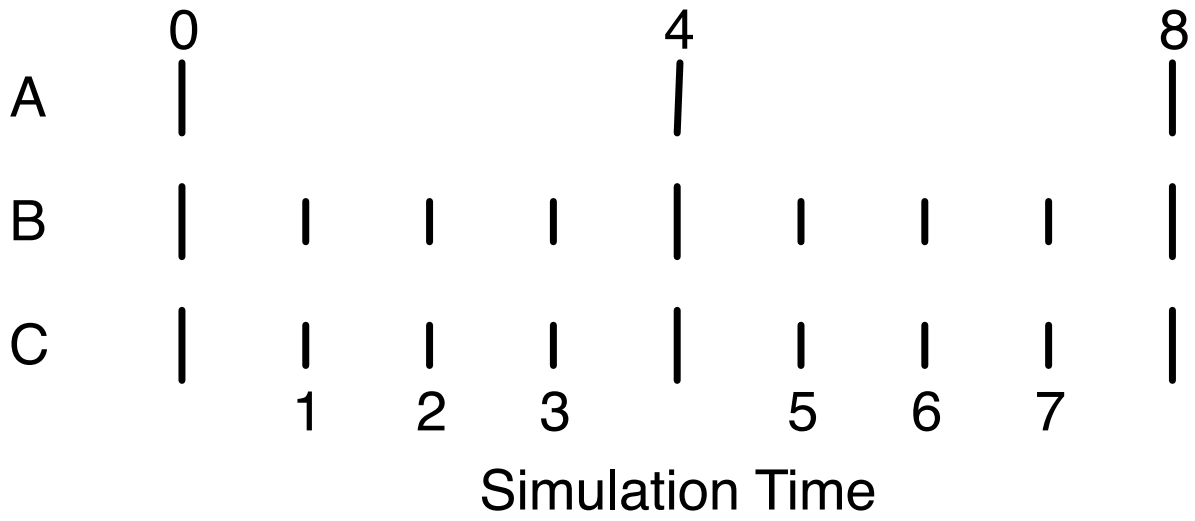


Why synchronization hurts, part II: Load balance



AMR Execution Overview

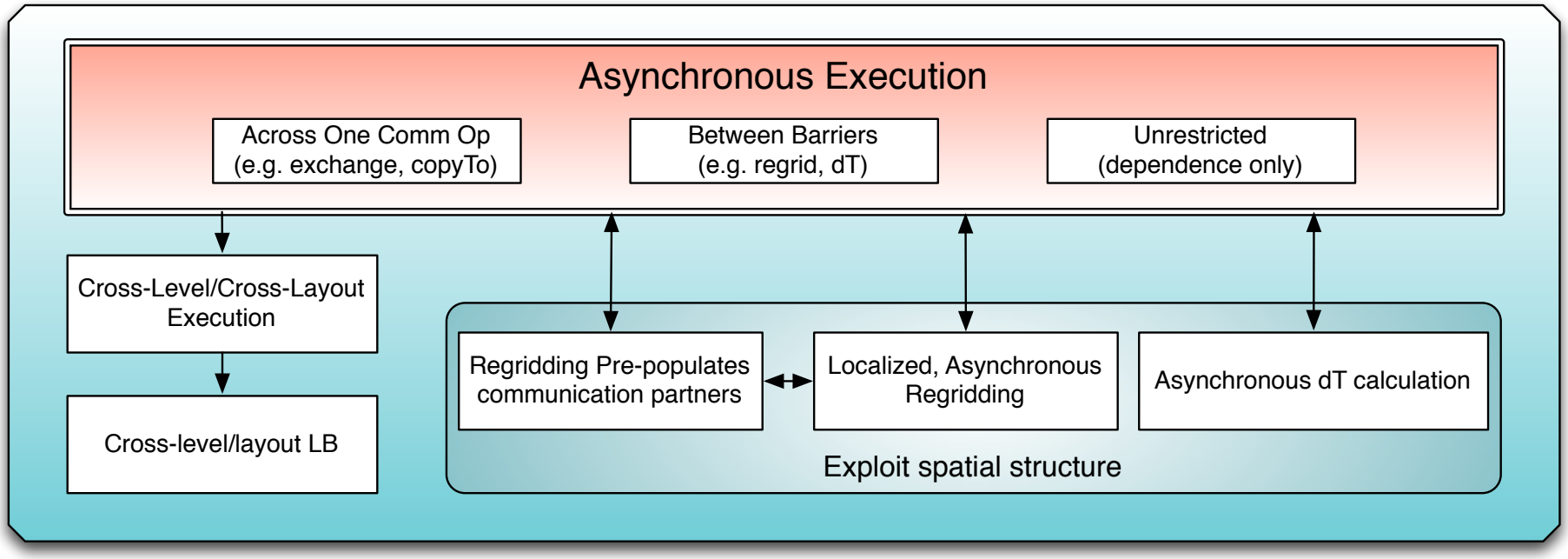




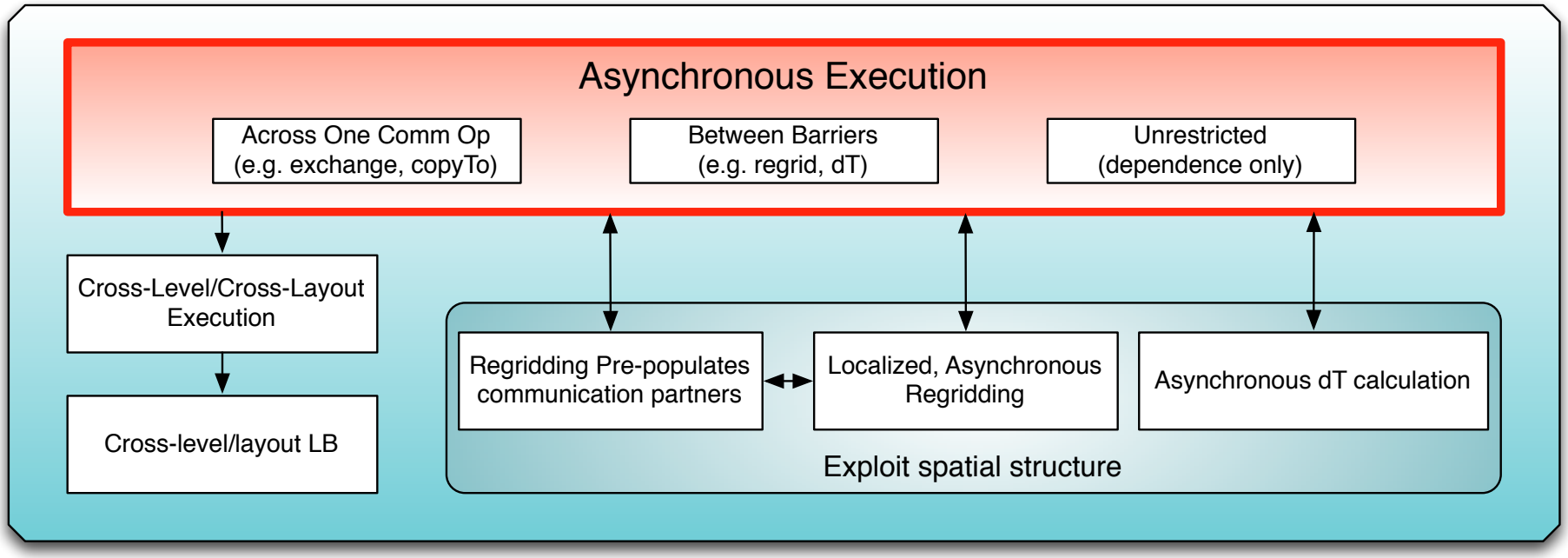
Synchronization points in AMR simulations

- Halo exchange
- Inter-level interpolation/averaging
- Timestep Calculation
- Domain Decomposition
- Elliptic Solvers

Overview


Asynchronous Execution
Across One Comm Op
(e.g. exchange, copyTo)
Between Barriers
(e.g. regrid, dT)
Unrestricted
(dependence only)
Cross-Level/Cross-Layout
Execution
Cross-level/layout LB
Regridding Pre-populates
communication partners
Localized, Asynchronous
Regridding
Asynchronous dT calculation
Exploit spatial structure

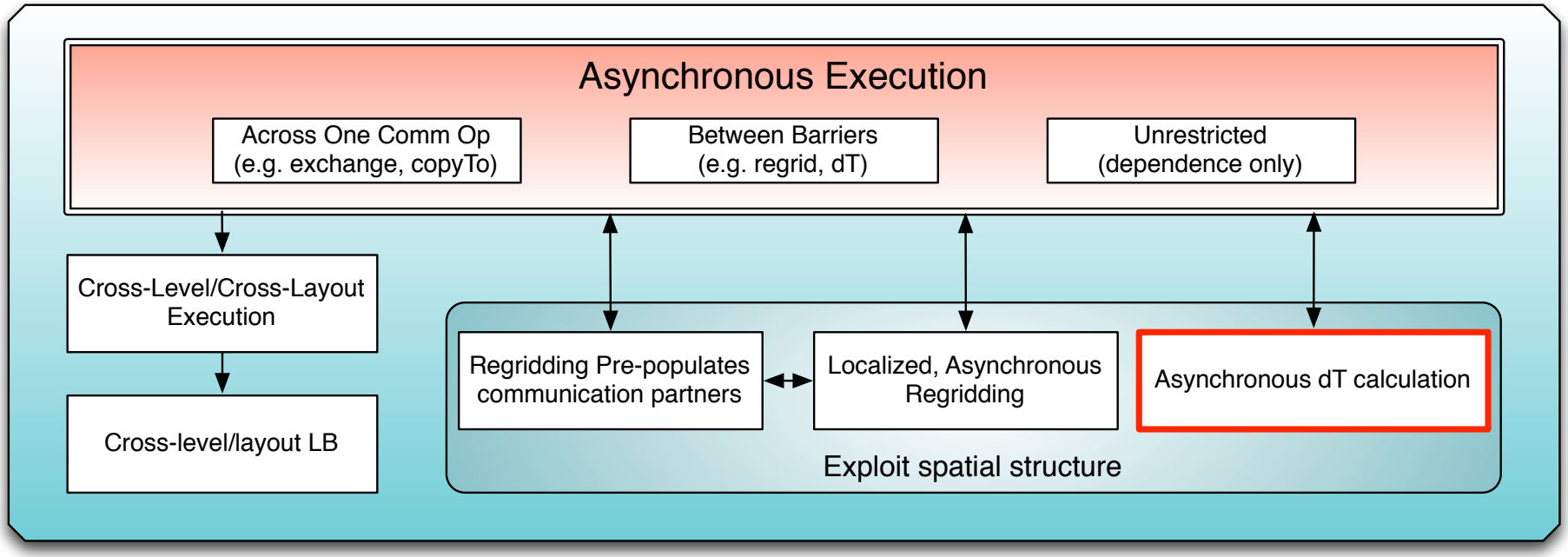
Asynchronous execution



Asynchronous execution

- Demonstrated in Uintah framework and Charm++ miniapp[1]
- Overlap computation with communication
- Execute in arbitrary order
- Extent can vary
- Challenging to adapt existing code
- Potential application of code transformation, e.g. via ROSE

[1] "Scalable Algorithms for Distributed-Memory Adaptive Mesh Refinement". Langer, Lifflander, **Miller**, Pan, Kale, Ricker. SBAC-PAD 2012


Asynchronous Execution
Across One Comm Op
(e.g. exchange, copyTo)
Between Barriers
(e.g. regrid, dT)
Unrestricted
(dependence only)
Cross-Level/Cross-Layout
Execution
Cross-level/layout LB
Regridding Pre-populates
communication partners
Localized, Asynchronous
Regridding
Asynchronous dT calculation
Exploit spatial structure

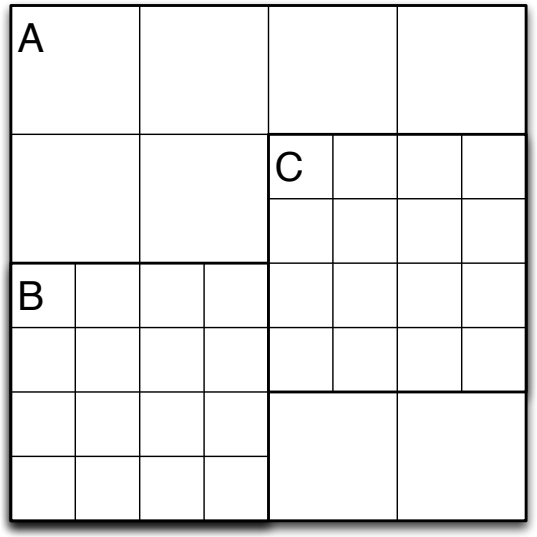
Localized Timestep Determination

- Maximum timestep determined by stability conditions - CFL, etc.
- Treated as global, though driven by local phenomena
- Computed globally via collective
- Subsequent steps depend on result - hard synchronization

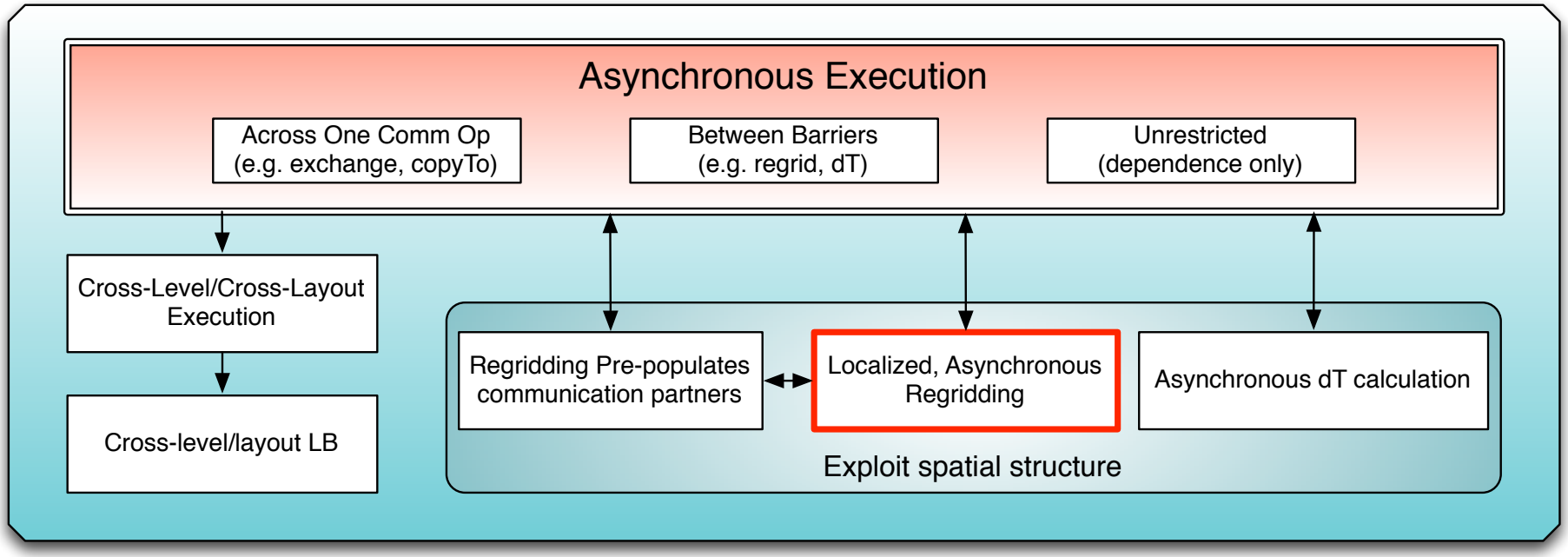
Localized Timestep Determination

- Maximum timestep determined by stability conditions - CFL, etc.
- Treated as global, though driven by local phenomena
- Computed globally via collective
- Subsequent steps depend on result - hard synchronization
- Compute on each block
- Coordinate with neighbors
- Interpolate as necessary

AMR Execution Overview



Refinement in space and time, independently


Asynchronous Execution
Across One Comm Op
(e.g. exchange, copyTo)
Between Barriers
(e.g. regrid, dT)
Unrestricted
(dependence only)
Cross-Level/Cross-Layout
Execution
Cross-level/layout LB
Regridding Pre-populates
communication partners
Localized, Asynchronous
Regridding
Asynchronous dT calculation
Exploit spatial structure

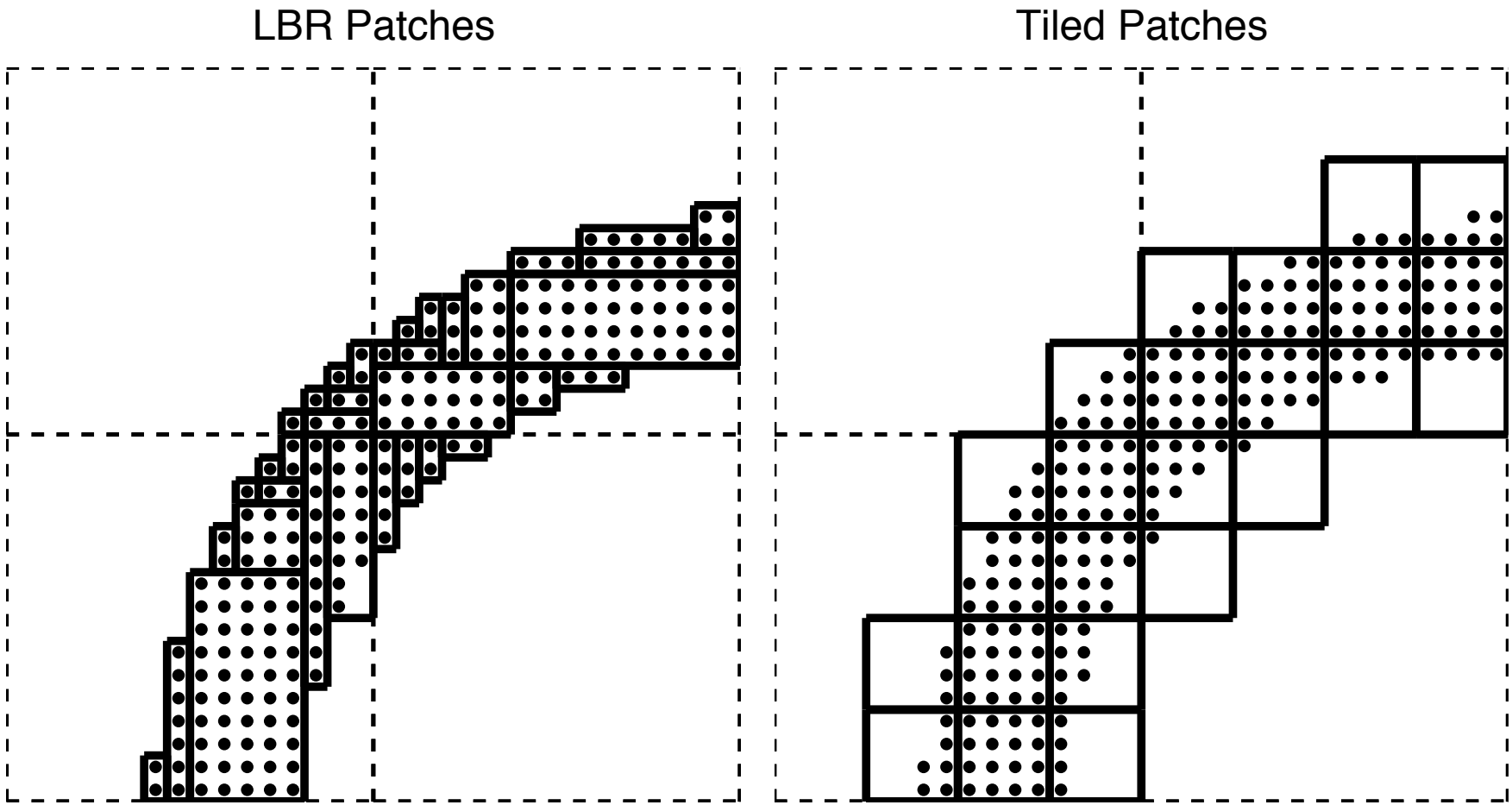
Domain Decomposition

- Set of blocks is 'global'
- Not every processor needs to know about every block
- Execute blocks independently, tell them each about their own neighbors
- Can run asynchronously, in parallel

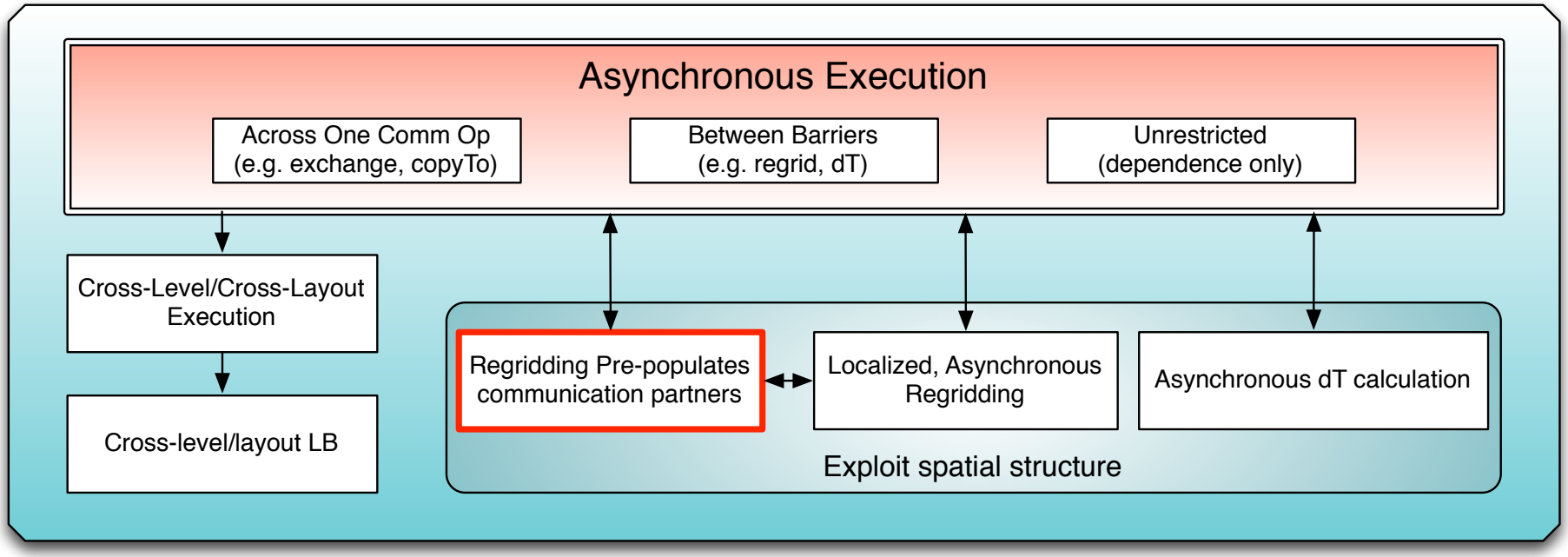
Domain Decomposition

- Set of blocks is ‘global’
- Not every processor needs to know about every block
- Execute blocks independently, tell them each about their own neighbors
- Can run asynchronously, in parallel
- Localized Berger-Rigoutsos clustering or Tiled decomposition
- Gunney’s ‘Bridge’ algorithm derives new neighbors from old

Domain decomposition: Box generation

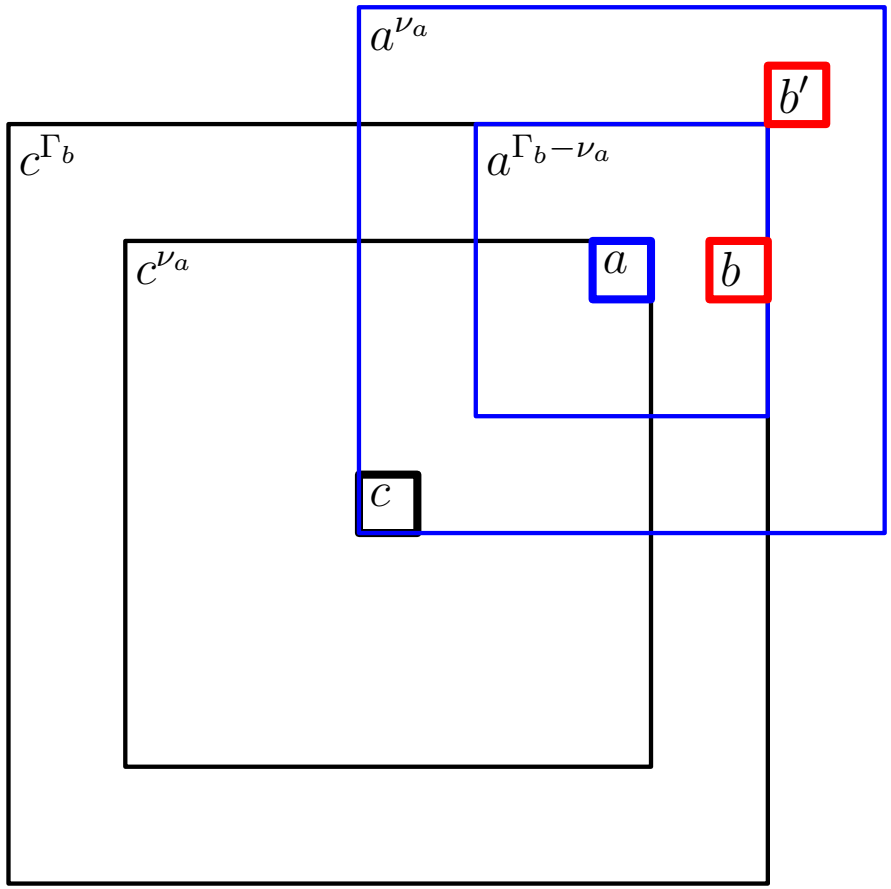


From Luitjens & Berzins 2011


Asynchronous Execution
Across One Comm Op
(e.g. exchange, copyTo)
Between Barriers
(e.g. regrid, dT)
Unrestricted
(dependence only)
Cross-Level/Cross-Layout
Execution
Cross-level/layout LB
Regridding Pre-populates
communication partners
Localized, Asynchronous
Regridding
Asynchronous dT calculation
Exploit spatial structure

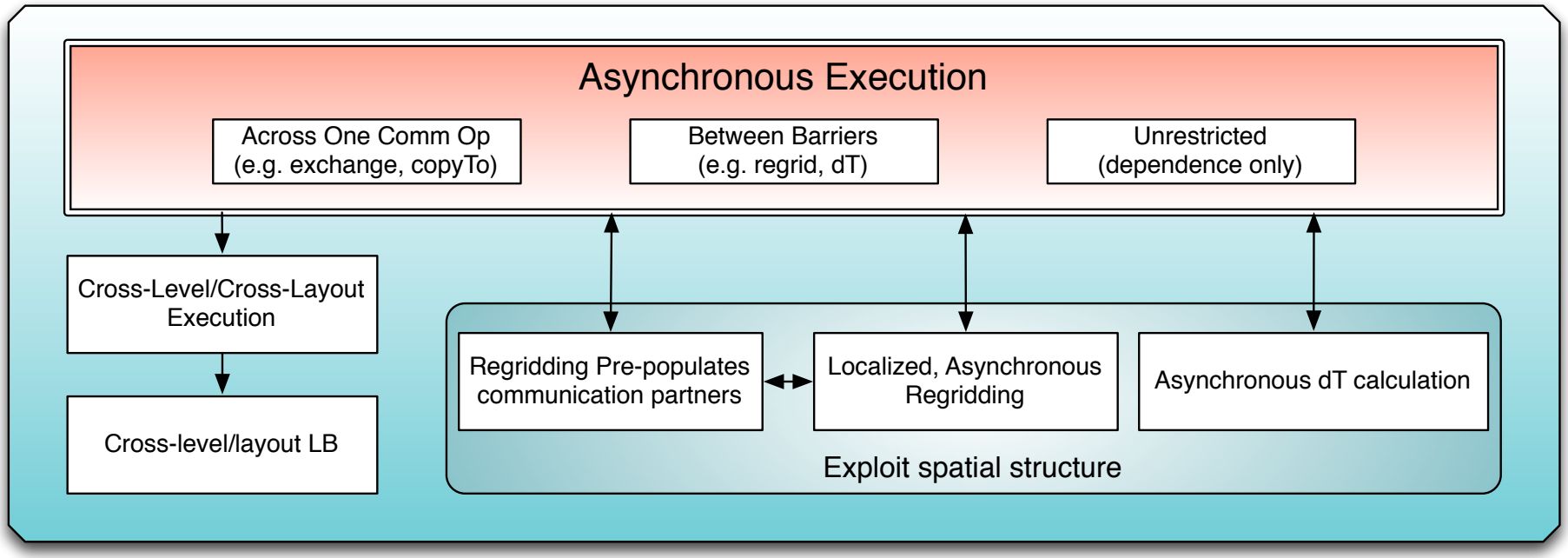
Domain Decompsition: neighbor determination

'Bridge' from A to B via C



Expected Impact

- Faster AMR simulations
- New optimizations for AMR developers to pursue
- New room for 'asynchronous iteration' methods
- Insights on asynchronous parallel algorithms



Tree-structured AMR Domain Decomposition Without Synchronization