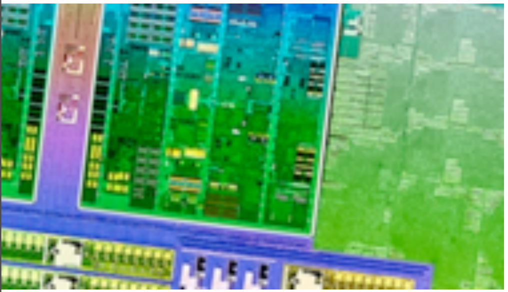
SAND2013-4813C

SAND2013-4868C



Proxy and Proto miniApps for Exascale Co-design

James A. Ang, and Richard F. Barrett
Scalable Computer Architectures Department
Sandia National Laboratories
Albuquerque, NM   USA

Sandia National Laboratories

Exceptional service in the national interest

International Supercomputing Conference
Leipzig, Germany
June 17-20, 2013

U.S. DEPARTMENT OF ENERGY | Office of Science

NNSA National Nuclear Security Administration

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Our Exascale Efforts Focus on Co-design

Our Exascale Efforts Focus on Co-design

- Key Co-Design capabilities
  - *SST* Architectural Simulation Framework
  - Pre-production, First-of-a-kind Testbeds
  - Scalable R&D System Software
  - *Mantevo* miniApplications

Our Exascale Efforts Focus on Co-design

- Key Co-Design capabilities
  - *SST* Architectural Simulation Framework
  - Pre-production, First-of-a-kind Testbeds
  - Scalable R&D System Software
  - *Mantevo* miniApplications
- Exascale Implications
  - Sustained commitment of significant funding
  - Requires the Long View >5 years out

    —> time to *influence* Hardware Architectures

Which is Harder to change: Hardware or Software?

Which is Harder to change: Hardware or Software?

- Conventional Wisdom
  - Hardware is difficult to change
  - Software is easy to change

Which is Harder to change: Hardware or Software?

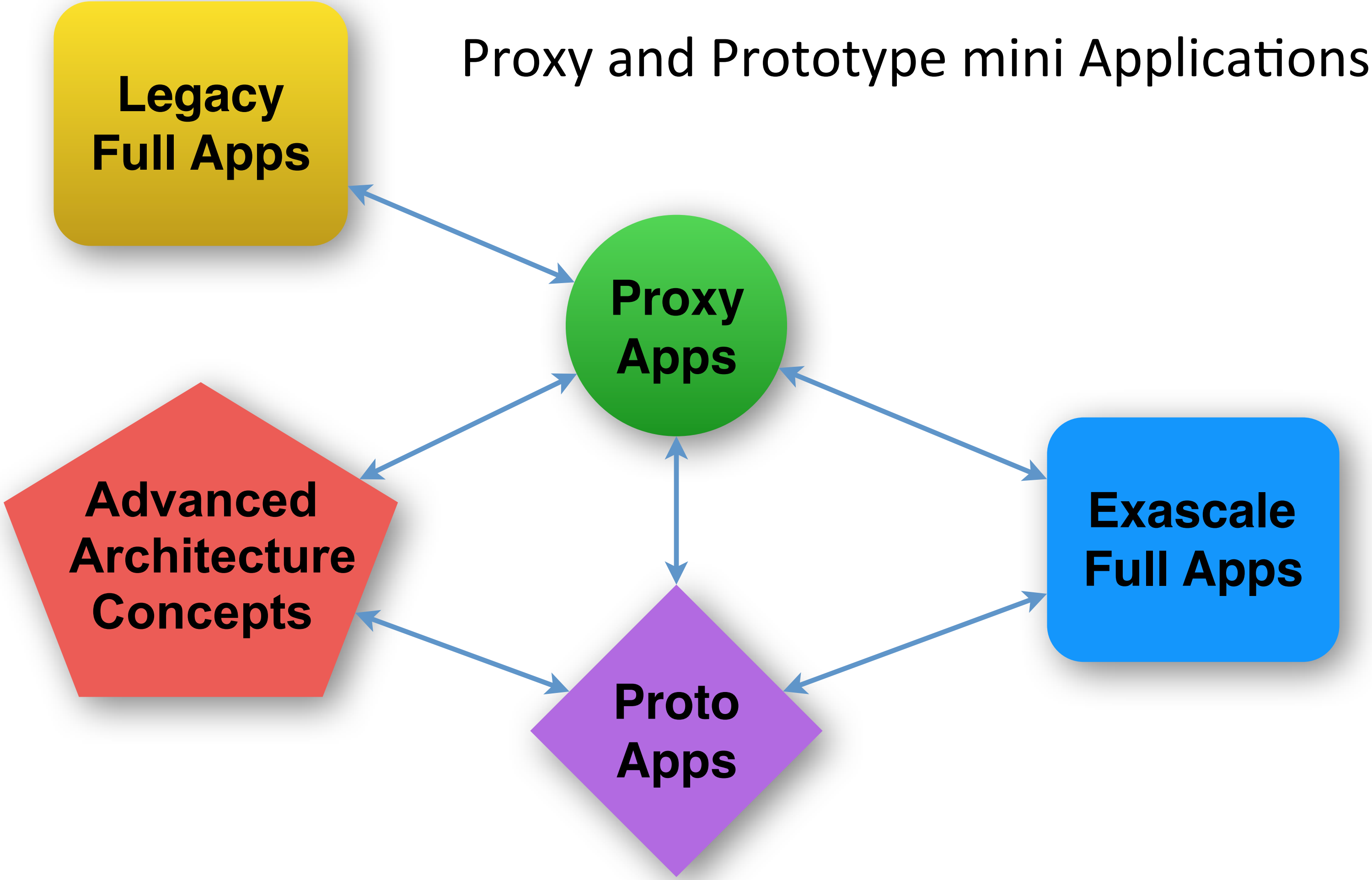
- Conventional Wisdom
  - Hardware is difficult to change
  - Software is easy to change
- For the Long View,

  Conventional Wisdom is wrong!

MiniApps: Proxy and Proto

- Relationship among Full Applications

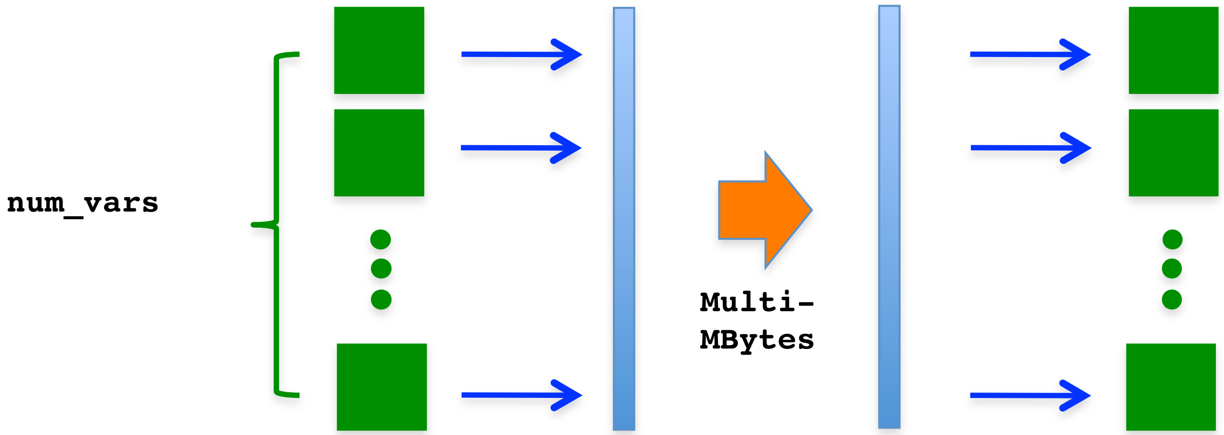
Proxy and Prototype mini Applications



Representative Legacy app: CTH

- Eulerian multi-material modeling application
- 3D, finite volume stencil computation
- BSP with message aggregation (BSPMA)

```
do i = 1, num_tsteps
```

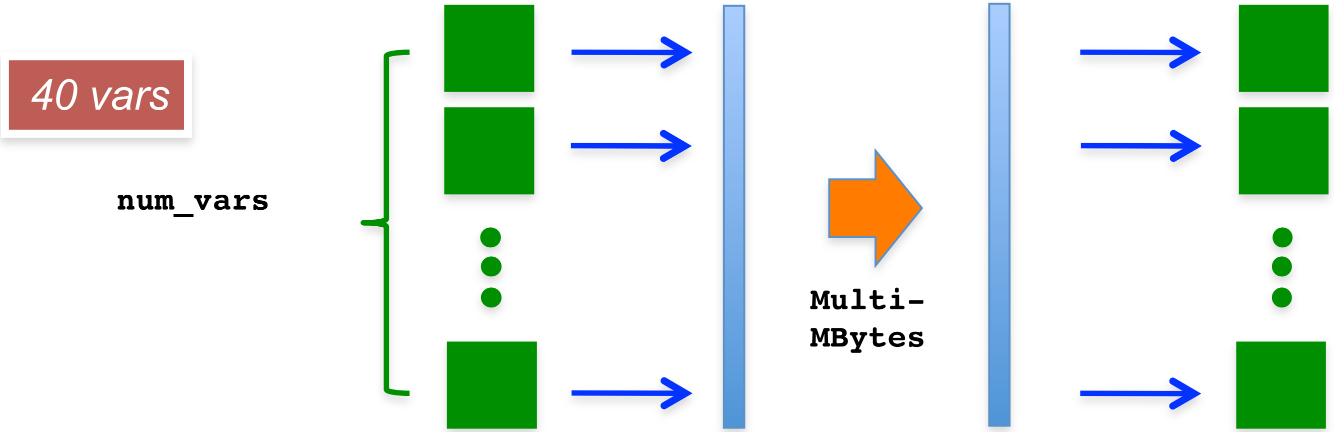


```
    do j = 1, num_vars
        compute
    end do
end do
```

Representative Legacy app: CTH

- Eulerian multi-material modeling application
- 3D, finite volume stencil computation
- BSP with message aggregation (BSPMA)

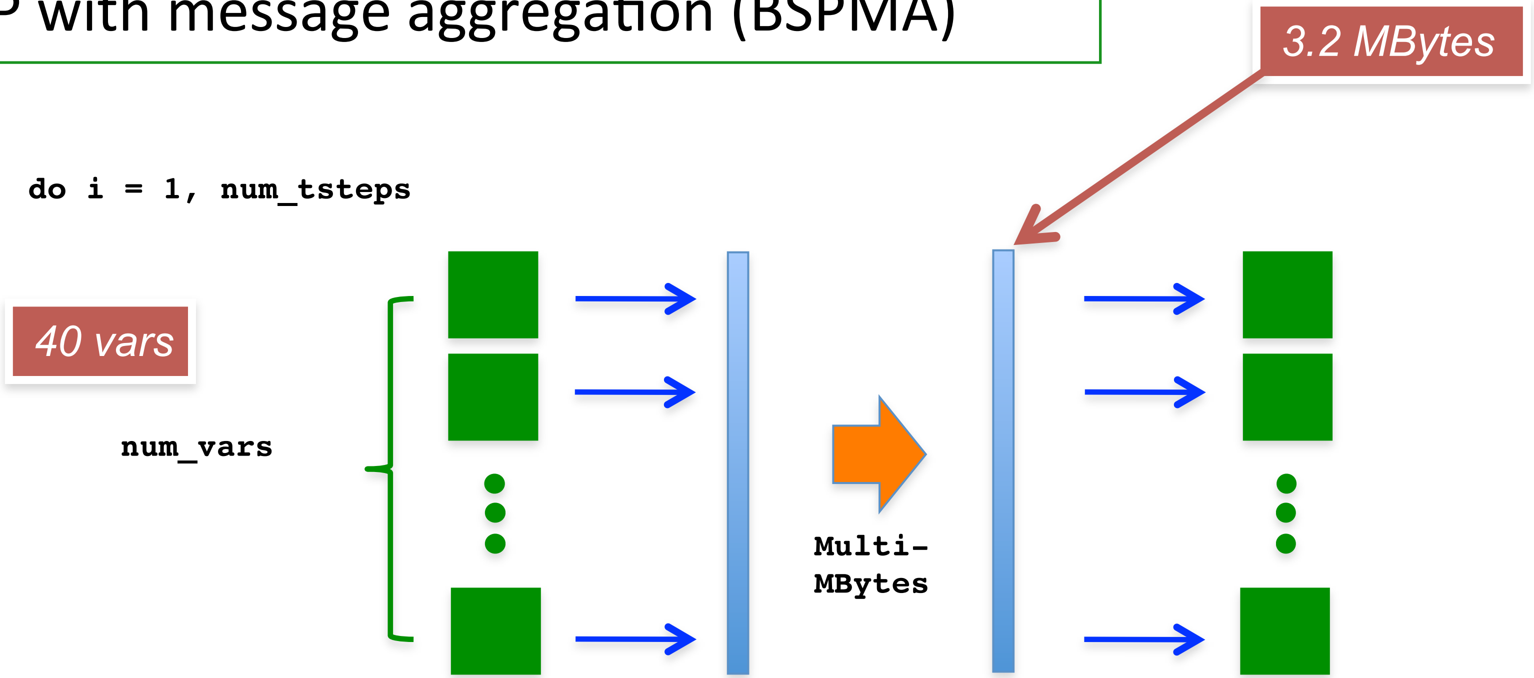
```
do i = 1, num_tsteps
    do j = 1, num_vars
        compute
    end do
end do
```



Representative Legacy app: CTH

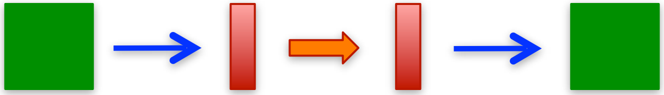
- Eulerian multi-material modeling application
- 3D, finite volume stencil computation
- BSP with message aggregation (BSPMA)

```
do i = 1, num_tsteps

    do j = 1, num_vars
       compute
    end do

end do
```



Alternative inter-node strategy: Single Variable Aggregated Faces (SVAF)

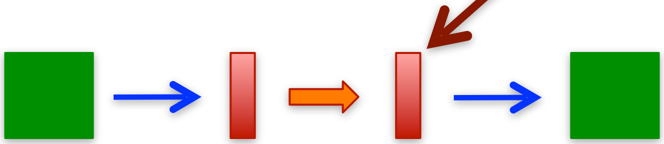
```
do i = 1, num_tsteps

  do j = 1, num_vars

     compute
```



```
  end do

end do
```

Alternative inter-node strategy: Single Variable Aggregated Faces (SVAF)

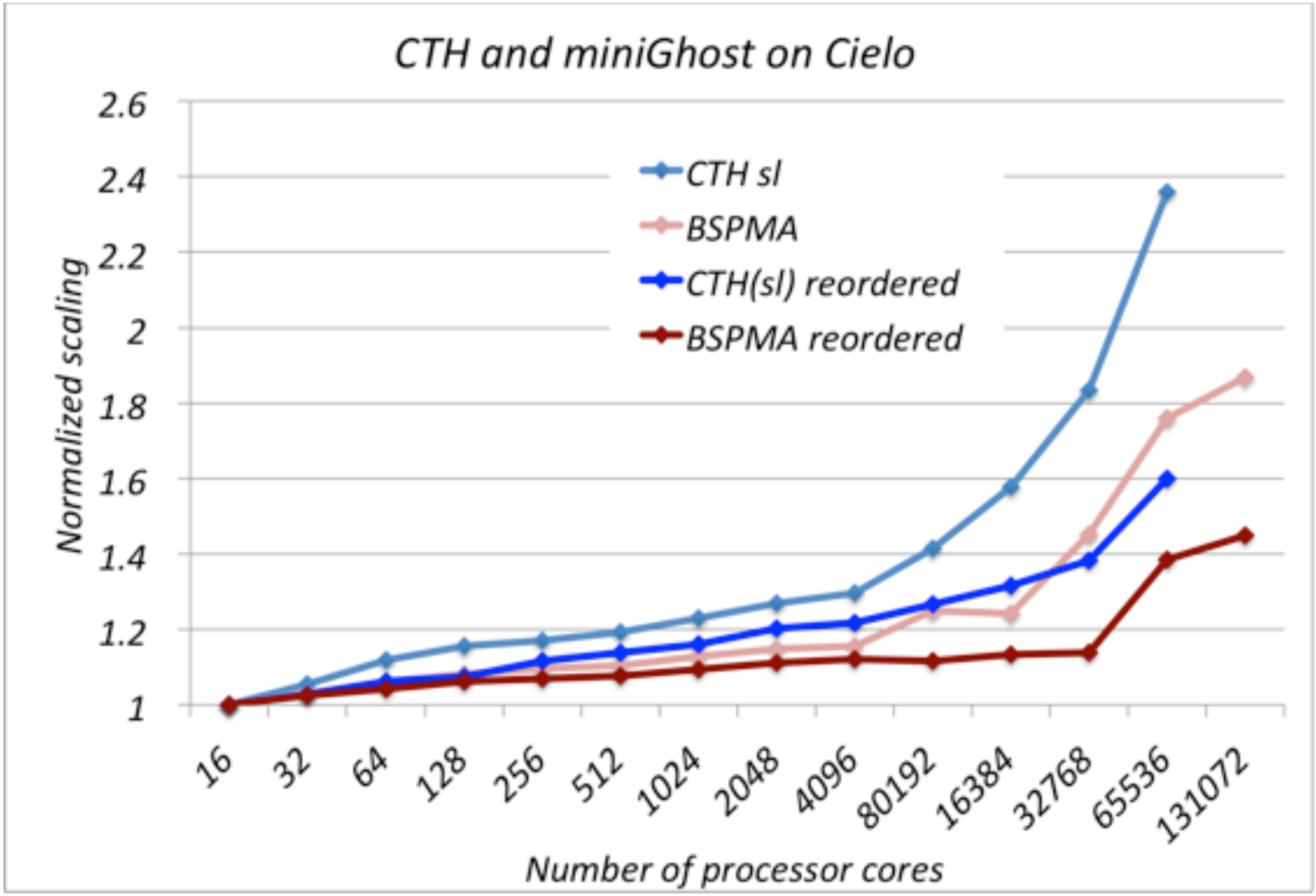
```
do i = 1, num_tsteps
 do j = 1, num_vars
   compute
```



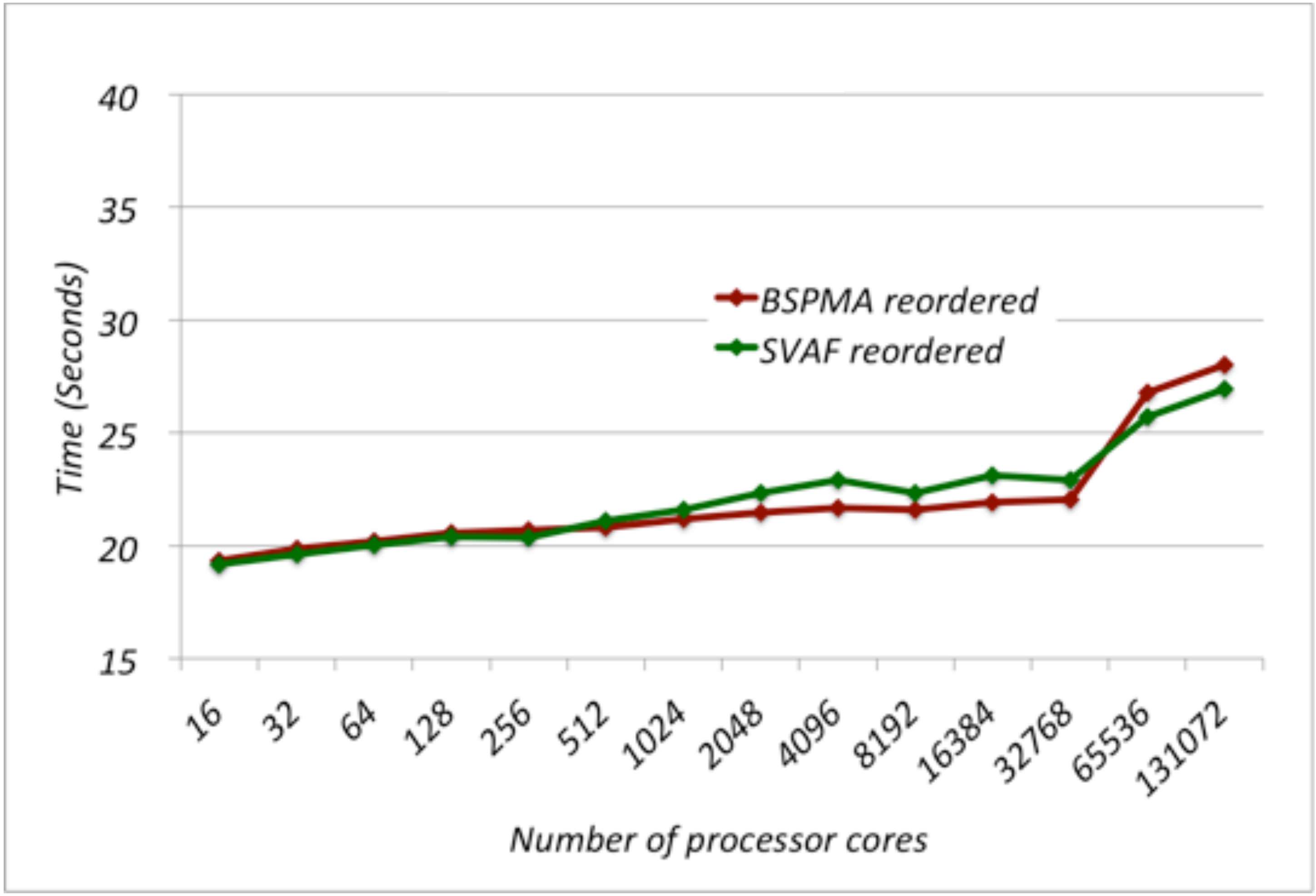
3.2 Mbytes / 40 = 80kbytes

```
 end do
end do
```

Physical ordering of MPI ranks



miniGhost on Cielo: BSPMA and SVAF



Implications for Co-design at a System Architecture Level

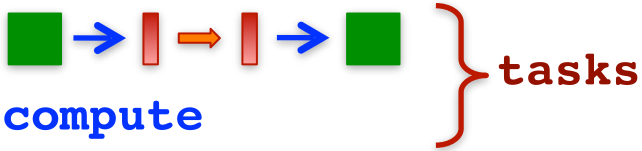
- We know that BSPMA applications create congestion problems for high radix topologies

- Hypothesis: If applications are task parallel, supported with asynchronous runtime software, and multithreaded processors, then congestion issues dissipate

- Focus area for our R&D with *Proto* miniApps
  - Application – Task parallel implementation
  - System Software – Support for asynchronous, adaptive threads
  - Node Architecture – Support for light-weight threading
  - System Architecture – Interconnect Fabric with high radix routers and high injection message rate

miniGhost task parallel

```
do i = 1, num_tsteps
 do j = 1, num_vars
   do iblk, jblk, kblk
```



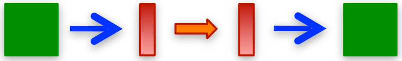
compute

tasks

```
   end do
 end do
end do
```

miniGhost task parallel

```
do i = 1, num_tsteps
  do j = 1, num_vars
     do iblk, jblk, kblk
```



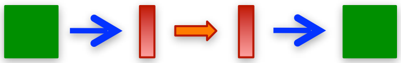
```
        compute                 tasks
     end do
  end do
end do
```

Compute *after* comm due to workspace usage model

miniGhost task parallel

Needs abstraction (DAG)

```
do i = 1, num_tsteps

  do j = 1, num_vars

     do iblk, jblk, kblk
```

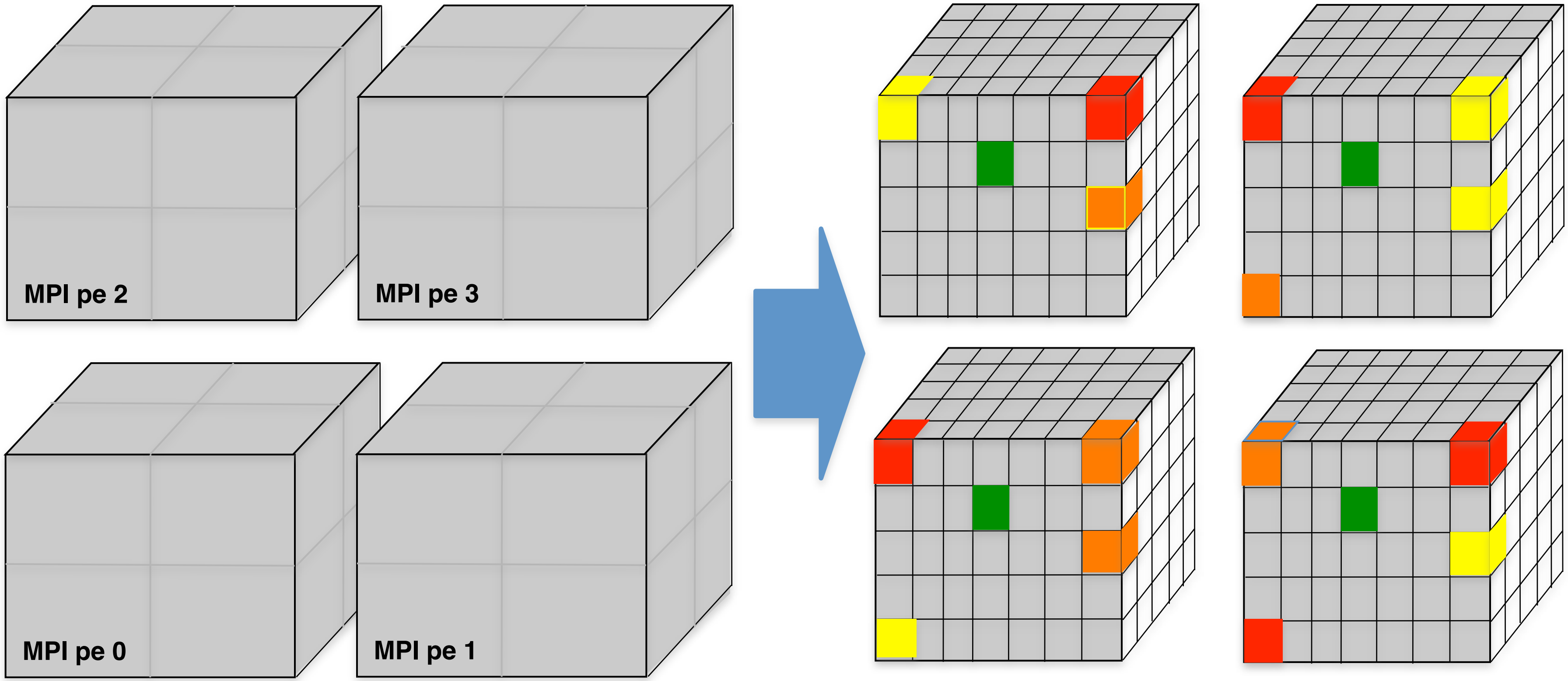


tasks

```
        compute

     end do

  end do

end do
```

Compute *after* comm
due to workspace
usage model

miniGhost: over-decomposition task parallel implementation



Data parallel

⬜ thread

Task parallel: some representative task workloads

🟩 Computation

🟧 Computation + MPI

🟨 Computation + BC

🟥 Computation + BC + MPI

miniGhost tp: Adding AMR

- Diffusion over the 3D domain with random initial conditions and reflective boundary conditions
  - Refinement based on the boundary or volume of an object being moved through the mesh and changing size. So, for example, a shock front can be simulated by refining based on a sphere which starts small and grows in size as the problem advances.

- Refinement within blocks
  - A block is refined into 8 blocks
  - Neighbors must be within one level of refinement
  - Computation is self-contained within a block

- Communication aggregated to BSP model
  - Excellent candidate for task parallelism version

Concluding Thoughts

Concluding Thoughts

- Multiple Dimensions of Co-Design
  - HW: Node and System architecture
  - SW: Application and System Software

Concluding Thoughts

- Multiple Dimensions of Co-Design
  - HW: Node and System architecture
  - SW: Application and System Software
- Examine *Conventional Wisdoms* on COTS and system balance
  - Component performance
  - Investment
    - Platform costs
    - R&D investments