Parallelization Strategies for High-order Discretized Hyperbolic PDEs

Hari Sundar, Jesse Kelley and Omar Ghattas

Institute for Computational Engineering & Sciences

THE UNIVERSITY OF TEXAS AT AUSTIN

siam

Computational Science & Engineering

Target:

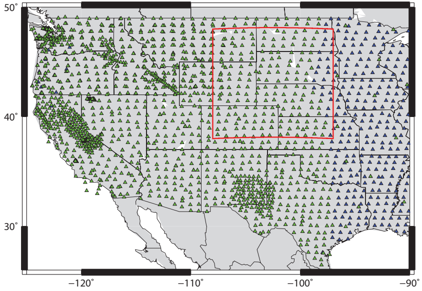
THE UNIVERSITY OF TEXAS AT AUSTIN

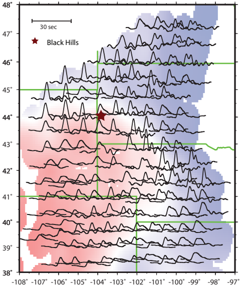
Inversion for local wave speed using full waveforms

- Solve and estimate the uncertainty in the solution for

$$\min_{\boldsymbol{m}} \mathcal{J}(\boldsymbol{m}) := \frac{1}{2} \int_0^T \int_B \mathcal{B}(\boldsymbol{x})(\boldsymbol{v} - \boldsymbol{v}_{data})^2 \, \boldsymbol{dx} + R(\boldsymbol{m})$$

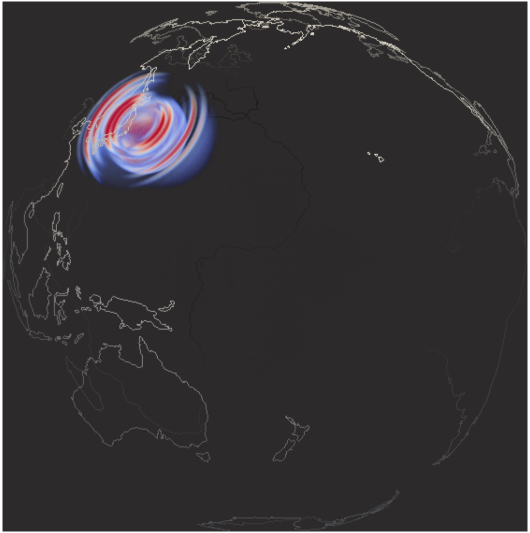
  where the elastic/acoustic wave equation maps the wave speeds $\boldsymbol{m} := (c_p(\boldsymbol{x}), c_s(\boldsymbol{x}))$ into $\boldsymbol{v}$
- $\boldsymbol{v}_{data}$ are waveform observations at receivers, R is a regularization/prior operator, and $\mathcal{B}(\boldsymbol{x})$ is an observation operator that reflects receiver locations and weights





Simulation of Japan earthquake

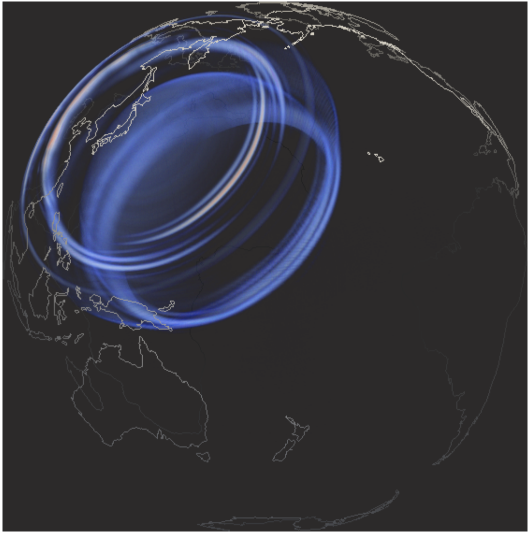
Using a simplified source representation



visualization by Greg Abram, TACC

Simulation of Japan earthquake

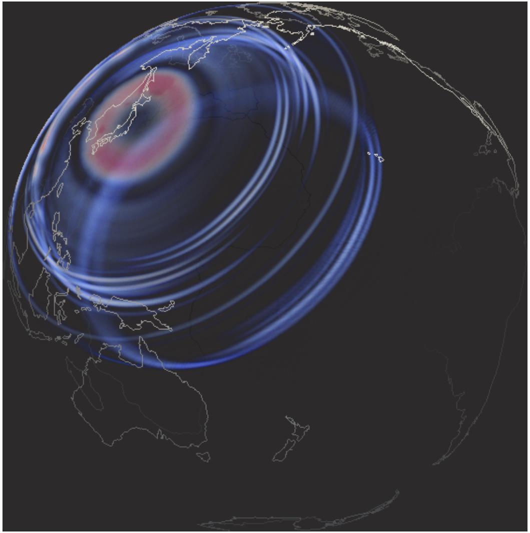
Using a simplified source representation



visualization by Greg Abram, TACC

Simulation of Japan earthquake

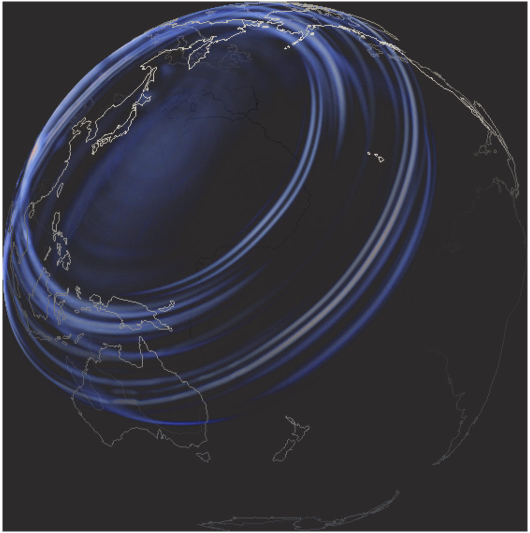
Using a simplified source representation



visualization by Greg Abram, TACC

Simulation of Japan earthquake

Using a simplified source representation



visualization by Greg Abram, TACC

Elastic/acoustic time-dependent wave equation

Governing equations in velocity-strain form

$$
\begin{aligned}
\frac{\partial \boldsymbol{E}}{\partial t} &= \frac{1}{2}\left(\nabla \boldsymbol{v} + \nabla \boldsymbol{v}^T\right) && \text{in } B \\
\rho \frac{\partial \boldsymbol{v}}{\partial t} &= \nabla \cdot (\mathbf{C}\boldsymbol{E}) + \boldsymbol{f} && \text{in } B \\
\boldsymbol{S}\boldsymbol{n} &= \boldsymbol{t}^{\text{bc}}(t) && \text{on } \partial B \\
\boldsymbol{v} &= \boldsymbol{v}_0(\boldsymbol{x}) && \text{at } t = 0 \\
\boldsymbol{E} &= \boldsymbol{E}_0(\boldsymbol{x}) && \text{at } t = 0
\end{aligned}
$$

- $\boldsymbol{E}$ --- strain tensor
- $\boldsymbol{S}$ --- stress tensor
- ρ --- mass density
- $\boldsymbol{v}$ --- displacement velocity
- f --- body force
- $\mathbf{C}$ --- constitutive tensor
- $\boldsymbol{t}^{\text{bc}}$ --- traction bc
- $\boldsymbol{v}_0, \boldsymbol{E}_0$ --- initial conditions
- t --- time
- $\boldsymbol{x}$ --- point in the body
- B --- solution body

Discontinuous Galerkin discretization

General Form

The dG discretization of the elastic-acoustic wave equations is given by:

Find $(\mathbf{E}, \boldsymbol{v}) \in V$ such that for all elements D^e:

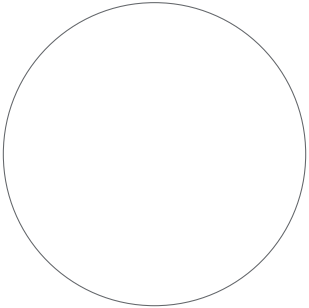
$$\int_{\mathsf{D}^e} \frac{\partial \boldsymbol{E}}{\partial t} : \boldsymbol{CH}\, d\boldsymbol{x} + \int_{\mathsf{D}^e} \rho \frac{\partial \boldsymbol{v}}{\partial t} \cdot \boldsymbol{w}\, d\boldsymbol{x} - \int_{\mathsf{D}^e} \frac{1}{2}\left(\nabla \boldsymbol{v} + \nabla \boldsymbol{v}^T\right) : \boldsymbol{CH}\, d\boldsymbol{x}$$
$$- \int_{\mathsf{D}^e} \left(\nabla \cdot (\boldsymbol{CE}) + \boldsymbol{f}\right) \cdot \boldsymbol{w}\, d\boldsymbol{x} + \int_{\partial \mathsf{D}^e} \mathfrak{F}_{\boldsymbol{v}} : \boldsymbol{CH} + \mathfrak{F}_{\boldsymbol{E}} \cdot \boldsymbol{w}\, d\boldsymbol{x} = 0$$

for all test functions $(\boldsymbol{H}, \boldsymbol{w})$,where $\mathfrak{F}_{\boldsymbol{v}}$ and $\mathfrak{F}_{\boldsymbol{E}}$ are the numerical fluxes.

$\Rightarrow$ Compute the numerical flux by solving the Riemann problem with discontinuous material parameters

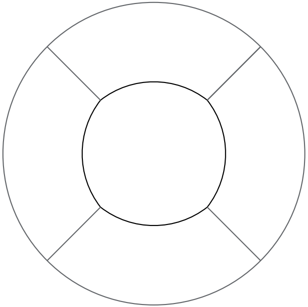
Meshing

forest of Octrees



Meshing

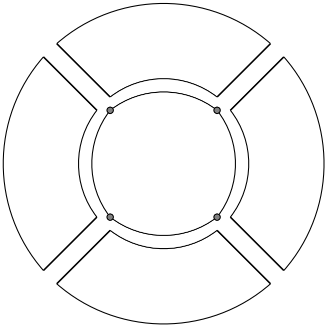
forest of Octrees



macromesh

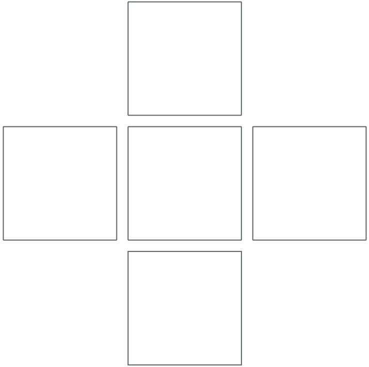
Meshing

forest of Octrees



Meshing

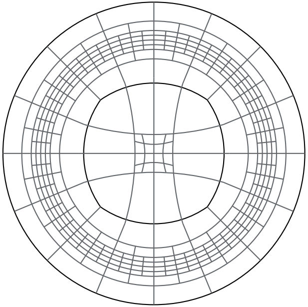
forest of Octrees



forest of octrees

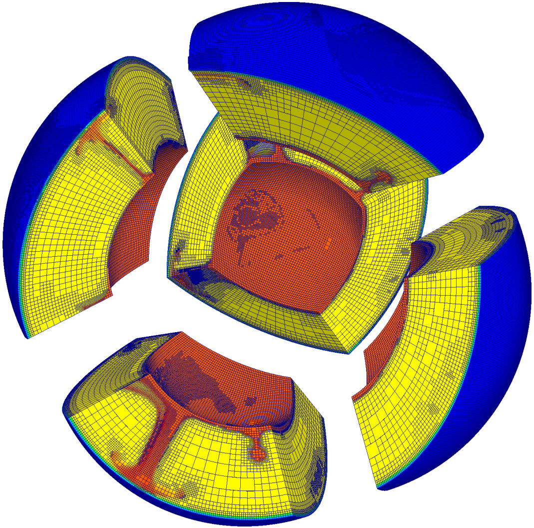
Meshing

forest of Octrees



Meshing

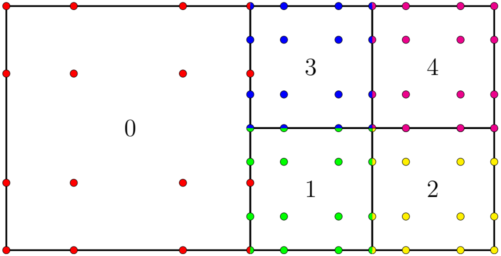
forest of Octrees



Discontinuous spectral element

THE UNIVERSITY OF TEXAS AT AUSTIN

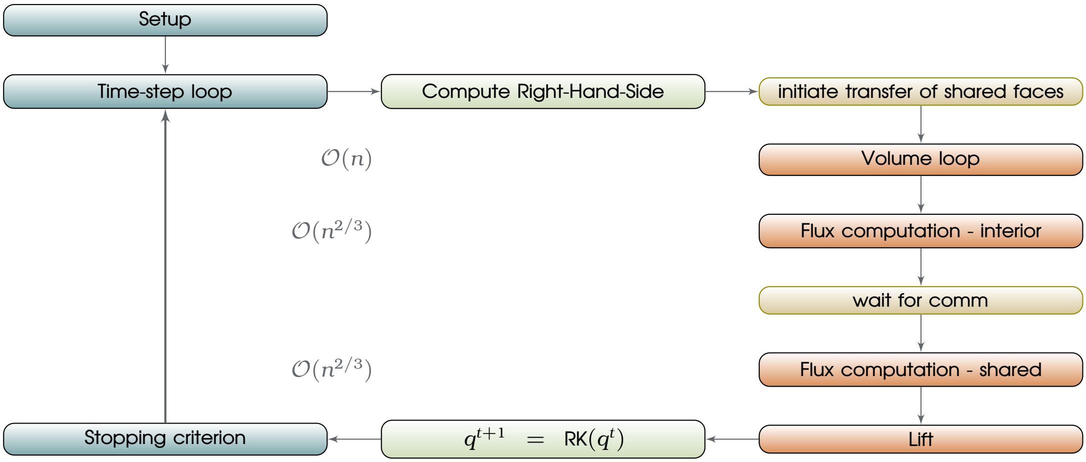
implementation



- allows h-nonconforming hexahedral elements (2:1 balance)
- the element basis is the tensor product of Lagrange polynomials based on the Legendre-Gauss-Lobatto (LGL) nodes (fast implementation)
- LGL quadrature for integration implies diagonal mass matrix
- allow material parameters to vary on element
- careful implementation of flux is required on hanging faces
- time discretization through method-of-lines (use RK4 in time)

Discontinuous spectral element implementation

pseudocode



Volume loop

tensor products

d-dimensional Tensor of size M

IIA

for $i \leftarrow 1$ ***to*** $M \times M$ **do**
  A(M*i);
  for $j \leftarrow 1$ ***to*** M **do**
    A(M*j+i);
  end
  for $k \leftarrow 1$ ***to*** M **do**
    A(M*k+i);
    for $j \leftarrow 1$ ***to*** M **do**
      A(M*i+j);
    end
  end
end

Parallelization Challenges

nested four level parallelism

- MPI - Distributed Memory
- CPU - MIC - (NUMA)
- OpenMP - Shared Memory Parallelism
- SIMD - Vectorization

Parallelization Challenges

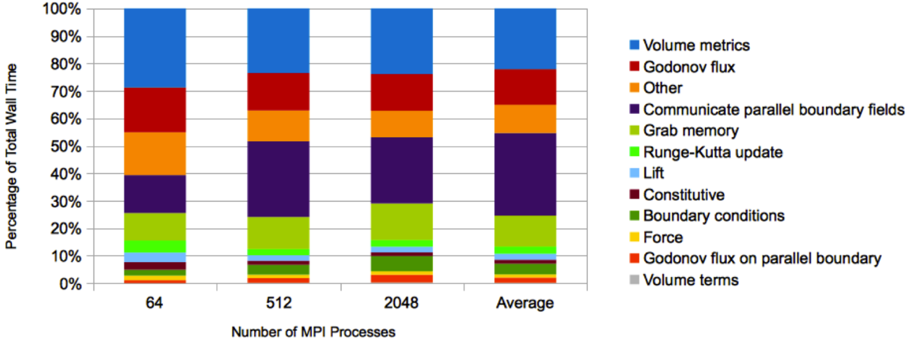
nested four level parallelism

- MPI - Distributed Memory
- CPU - MIC - (NUMA)
- OpenMP - Shared Memory Parallelism
- SIMD - Vectorization

need to parallelize on all four levels

Baseline Profiling

determine which stages to focus on



Optimizing for Stampede

I: Vectorization

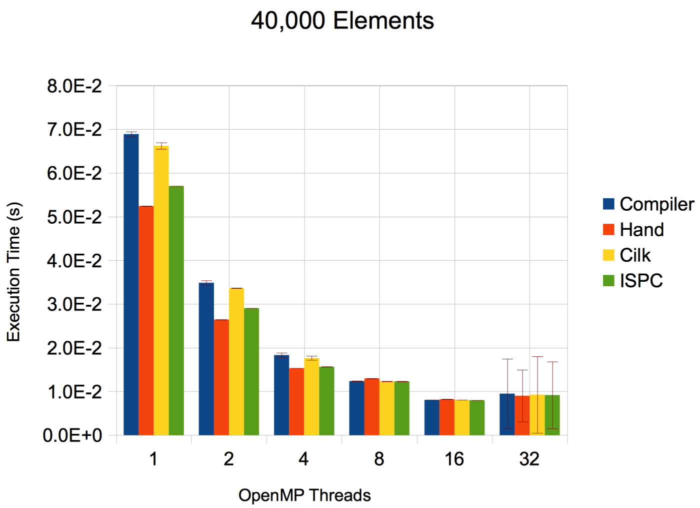
Vectorizing options,

- Hand-code separately for `avx` and `larrabee`
- Intel SPMD Program Compiler (`ispc`)
- Intel Cilk (`#pragma simd vectorlength(8)`)
- Compiler vectorization

Optimizing for Stampede

Vectorization + OpenMP

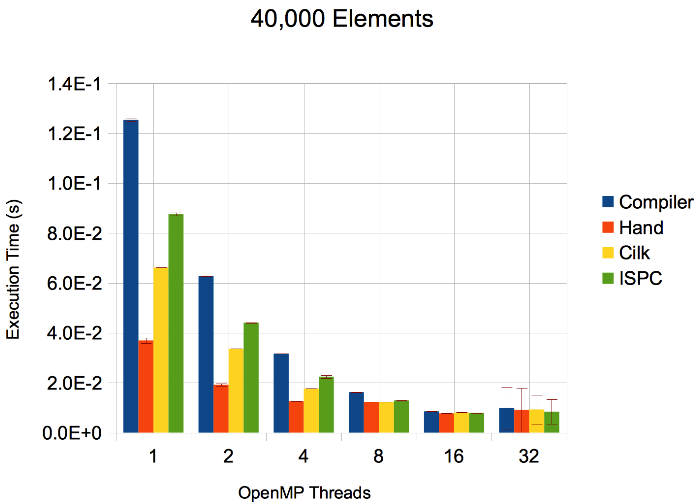
AIIX



Optimizing for Stampede

Vectorization + OpenMP

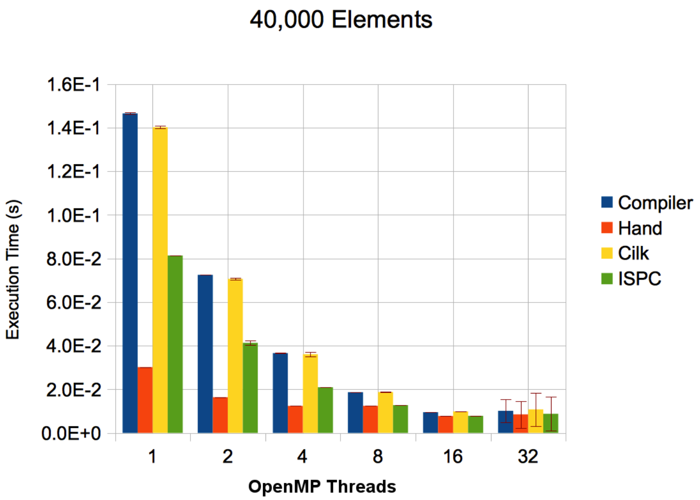
IAIX



Optimizing for Stampede

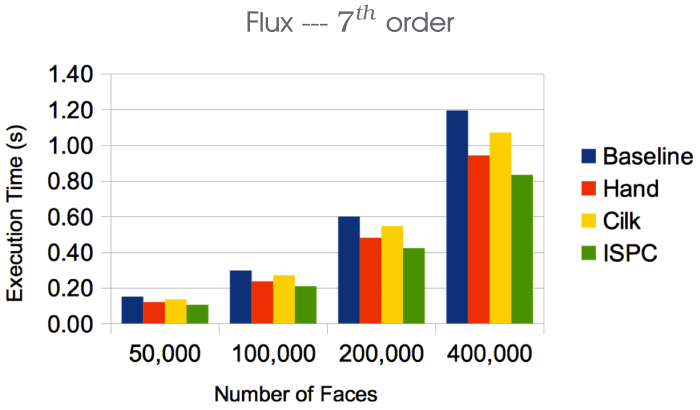
Vectorization + OpenMP

IIAX



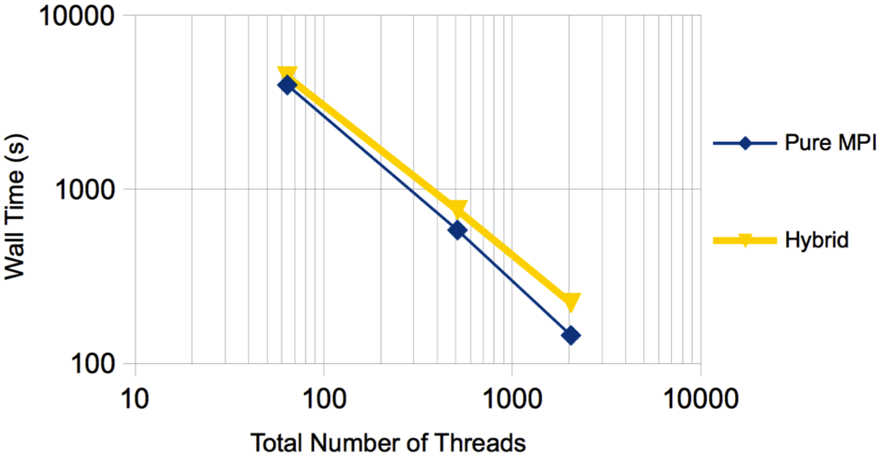
Optimizing for Stampede

Vectorization + OpenMP



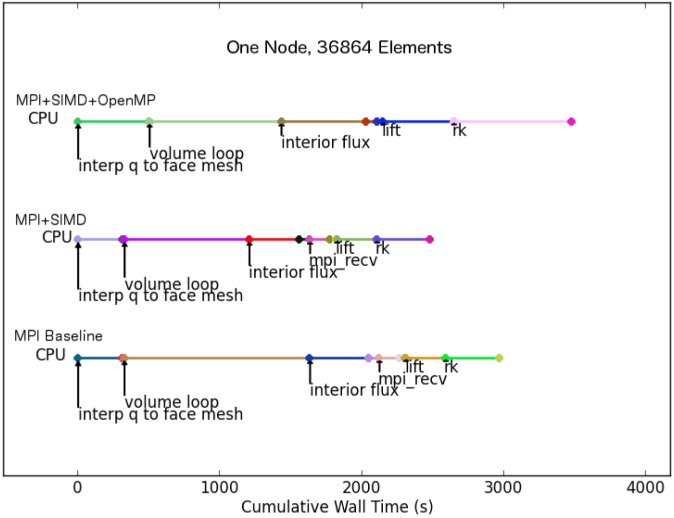
Optimizing for Stampede

MPI or MPI-OpenMP



Optimizing for Stampede

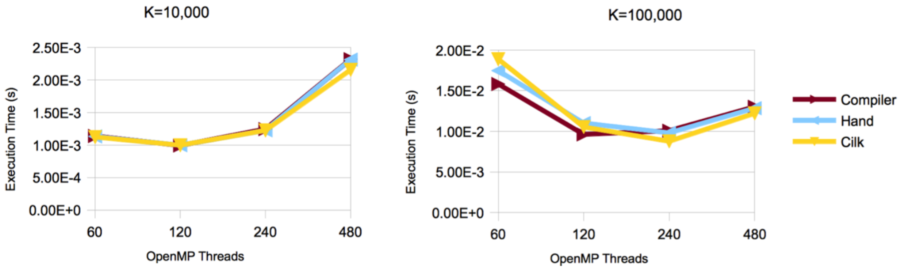
MPI or MPI-OpenMP



Optimizing for Stampede : MIC

Vectorization + OpenMP

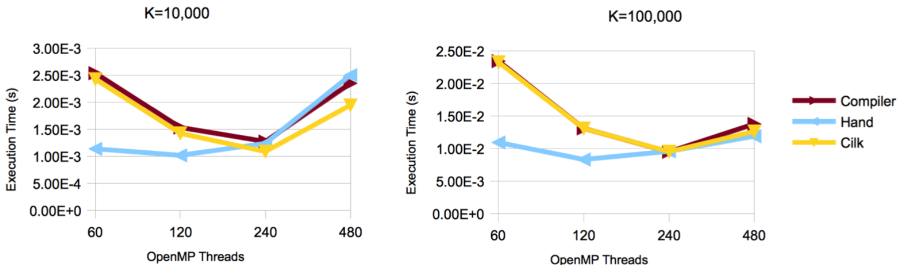
AIIX



Optimizing for Stampede : MIC

Vectorization + OpenMP

IAIX

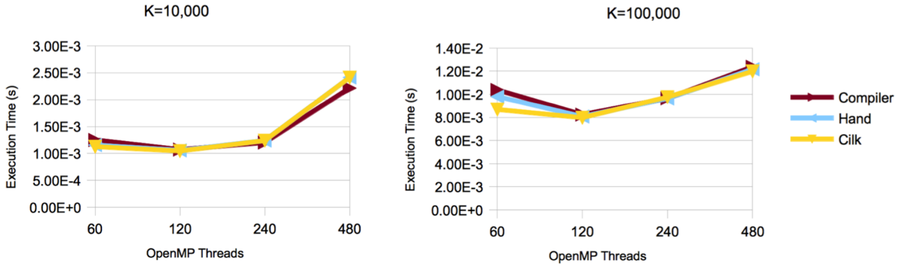


Optimizing for Stampede : MIC

THE UNIVERSITY OF TEXAS AT AUSTIN

Vectorization + OpenMP

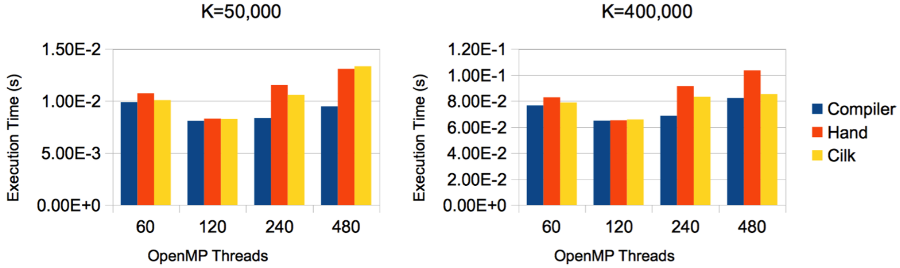
IIAX



Optimizing for Stampede : MIC

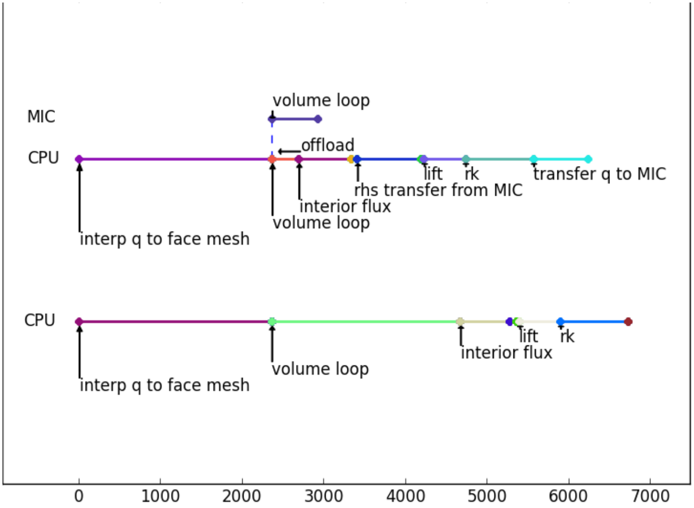
Vectorization + OpenMP

Flux --- 7^{th} order



CPU vs CPU+MIC

Offloading tasks



70% of Volume loop offloaded to MIC

Parallelizing on Stampede

lessons learnt

THE UNIVERSITY OF TEXAS AT AUSTIN

Available options

- task parallelism - Volume on MIC, flux on CPU
  - Communication $\propto N$ (as opposed to $N^{2/3}$)
- treat MIC same as another process
  - communication has to be routed through CPU
- Limit communications to CPU-CPU and CPU-MIC

Parallelizing on Stampede

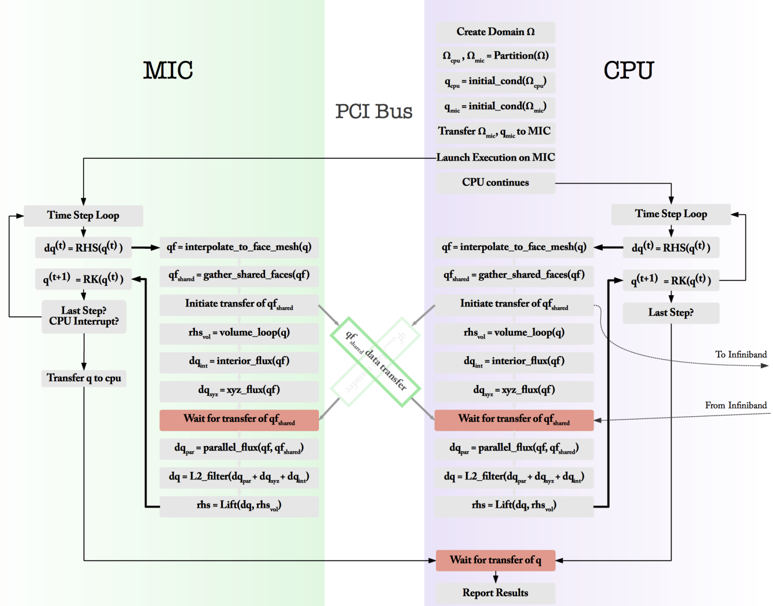
lessons learnt

Available options

- task parallelism - Volume on MIC, flux on CPU
  - Communication $\propto N$ (as opposed to $N^{2/3}$)
- treat MIC same as another process
  - communication has to be routed through CPU
- Limit communications to CPU-CPU and CPU-MIC
  - insulate MIC from other processes
  - need special partitioning

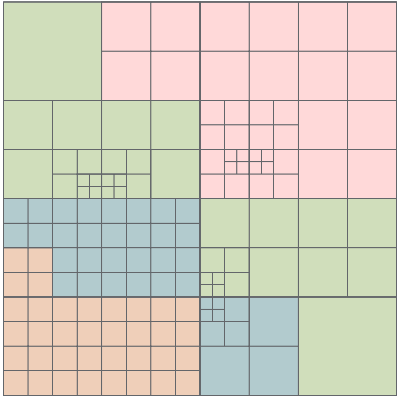
Modified Algorithm

Flowchart



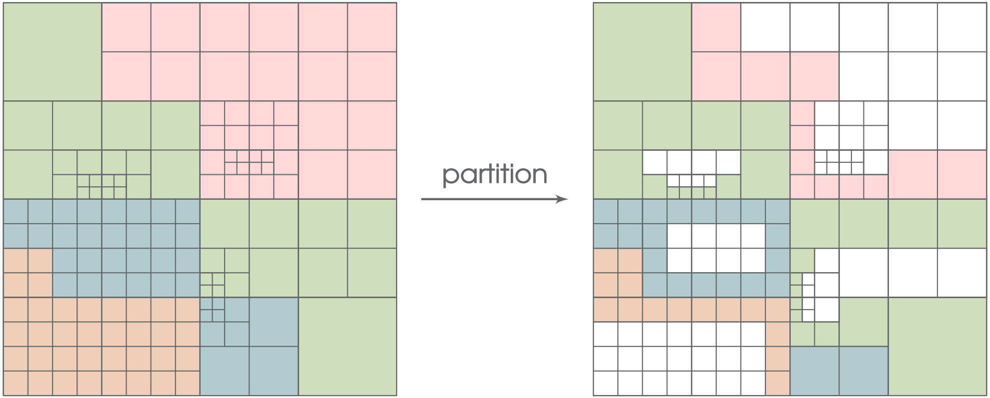
Modified Algorithm

needs special partitioning



Modified Algorithm

needs special partitioning



Conclusions

findings & future work

- Hand-vectorization works best,
- Limit MIC to perform only inter-node communication,
- Partitioning is hard, but can be done locally on each node.