Parallel Implementation of a Fast Marching solver for the Eikonal Equation

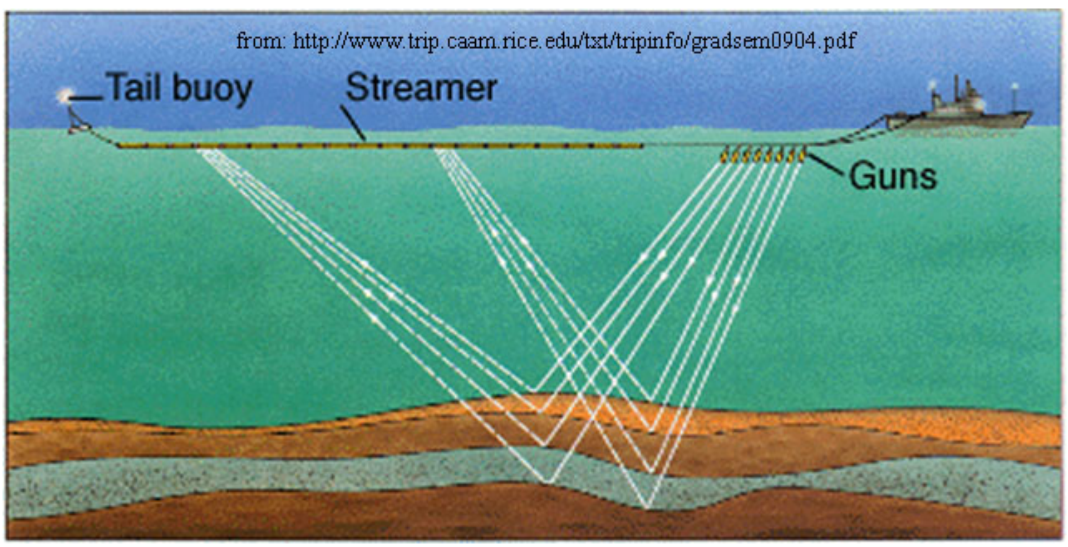
Leonardo Andrés Zepeda Núñez

Summary

- Seismic Imaging
- Eikonal Equation
- Viscosity Solution
- Numerical Methods
- Serial Implementation
- Parallel Implementation
- Conclusion
- Questions

Seismic Imaging

- Velocity Model $F(x,y)$
- Experimental Data $s(x,y=0)$
- First Arrival $T(x,y=0)$



Seismic Imaging

- Geometric Optics
- Ray Approximation
- Travel time function $T(x, y)$
- No Reflection
- Fitting on the surface

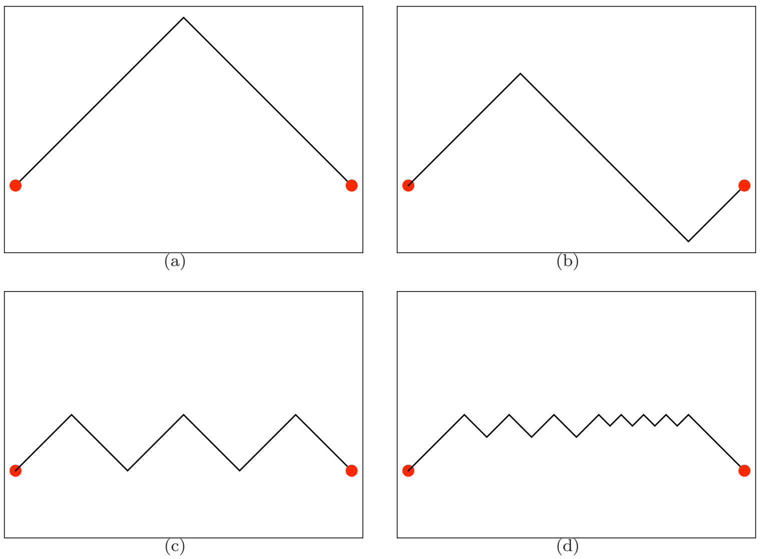
$$\min_F \|T_{\text{exp}}(x) - T_F(x, y = 0)\|_X$$

Eikonal Equation

- Equation $\|\nabla T(x,y)\| F(x,y) = 1$
- Distance function on a Manifold $g(\cdot,\cdot) = \dfrac{(\cdot,\cdot)}{v^2(x,y)}$
- Computation of the geodesic distance
- Solution is not unique

$$\begin{cases} |u'(x)| = 1 \ \text{in}\,(-1,1) \\ u(x) = 0, \quad x = \pm 1. \end{cases}$$

$$u(x) = 1 - |x|$$



Viscosity Solution

- Physical Solution
- Presence of Viscosity in real World

$$\|\nabla T_\epsilon(x,y)\| = \frac{1}{F(x,y)} + \epsilon \triangle T_\epsilon(x,y)$$

- Regularity and Limit $\lim_{\epsilon \to 0} T_\epsilon = T$

$$T_\epsilon \overset{?}{\to} T$$

Viscosity Solution

- Entropy
- Why do we care?
- Unique Solution
- Different Schemes won’t give the good answer

Numerical Method

- Discretization
- Grid
- Derivatives
- Upwind Methods

$$T_{i,j} = (x_i, y_j) = (i\Delta x, j\Delta y)$$

$$\frac{\partial T}{\partial x}_{i,j} \approx L(T_{i,j})$$

$$D^{x}_{i,j}T = \frac{T(x_{i+1}, y_j) - T(x_i, y_j)}{\Delta x}$$

$$D^{-x}_{i,j}T = \frac{T(x_i, y_j) - T(x_{i-1}, y_j)}{\Delta x}$$

$$D^{y}_{i,j}T = \frac{T(x_i, y_{j+1}) - T(x_i, y_j)}{\Delta y}$$

$$D^{-y}_{i,j}T = \frac{T(x_i, y_j) - T(x_i, y_{j-1})}{\Delta y}$$

Numerical Methods

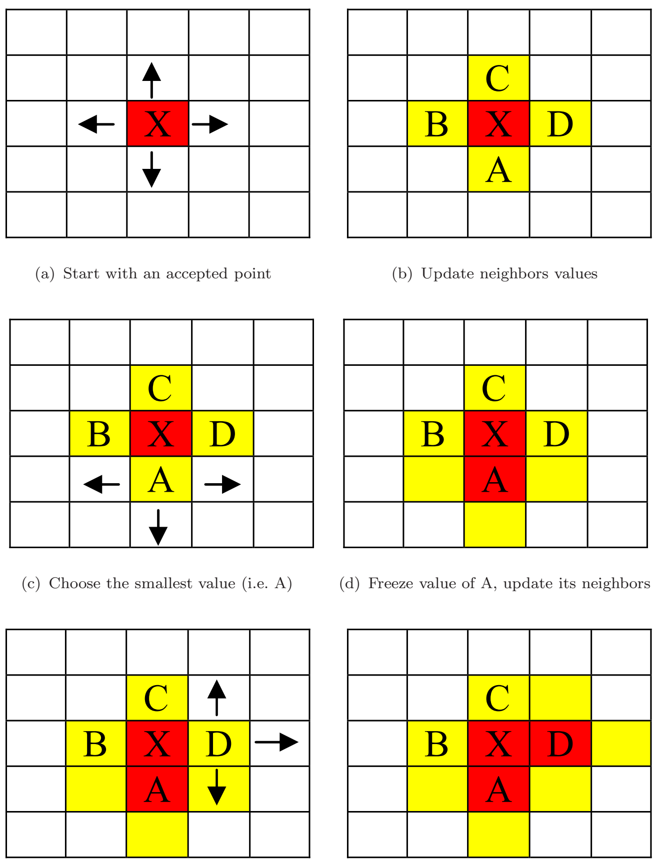
- Upwind Schemes

$$\Big(\max(D_{i,j}^{-x}T, 0)^2 + \min(D_{i,j}^{x}T, 0)^2 + \max(D_{i,j}^{-y}T, 0)^2 + \min(D_{i,j}^{y}T, 0)^2 \Big)^{\frac{1}{2}} = \frac{1}{F(x_i, y_j)}$$

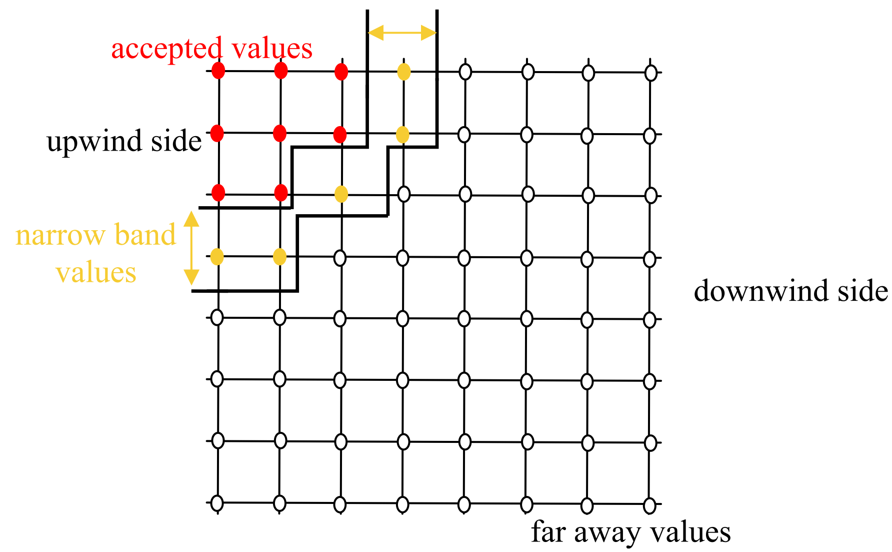
- Iterative solver
- Data Dependency

Numerical Method

- Fast Marching Method

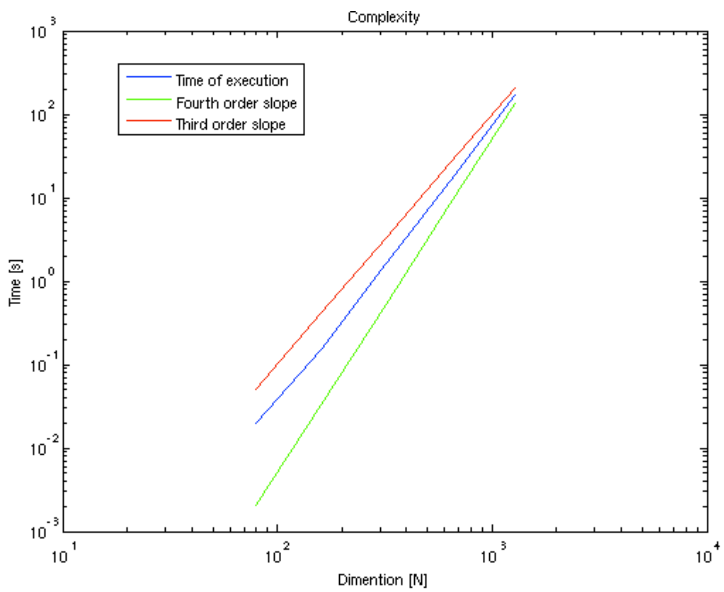


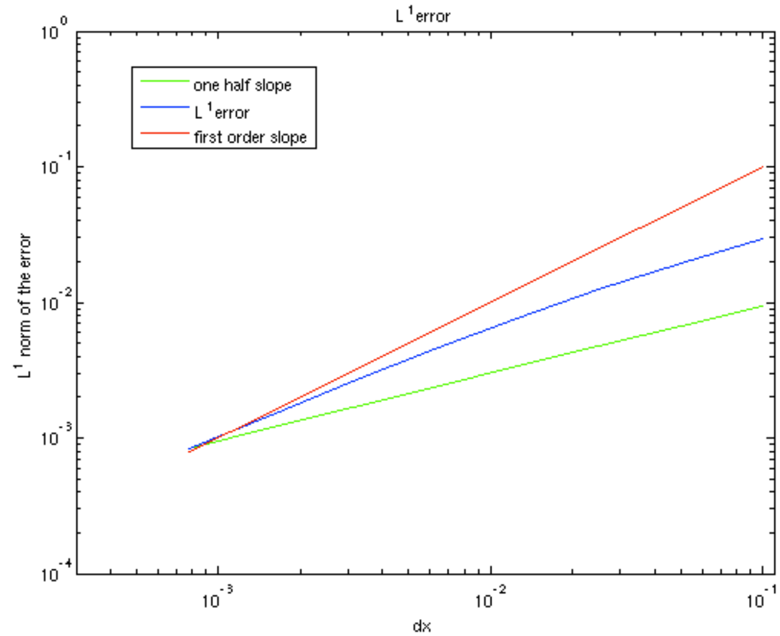
(a) Start with an accepted point
(b) Update neighbors values
(c) Choose the smallest value (i.e. A)
(d) Freeze value of A, update its neighbors
(e) Choose the smallest value (i.e. D)
(f) Freeze value of D, update its neighbors



Sequential Implementation

- Fast Marching demo
- Convergence
- Complexity



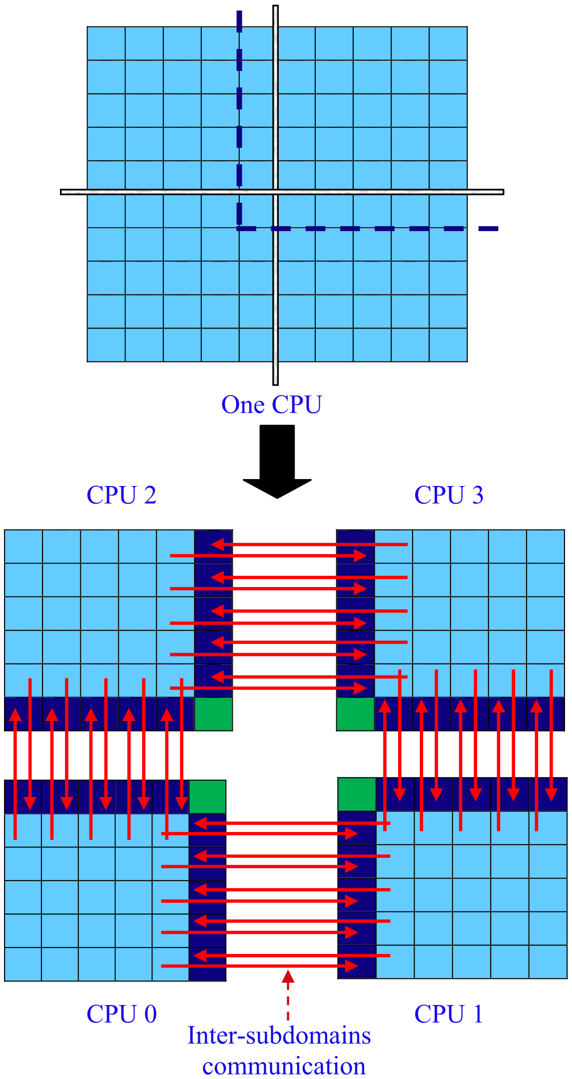


$$C(N) = N^{3,3462}$$

$$C_{optimal}(N) = N^3 \log N$$

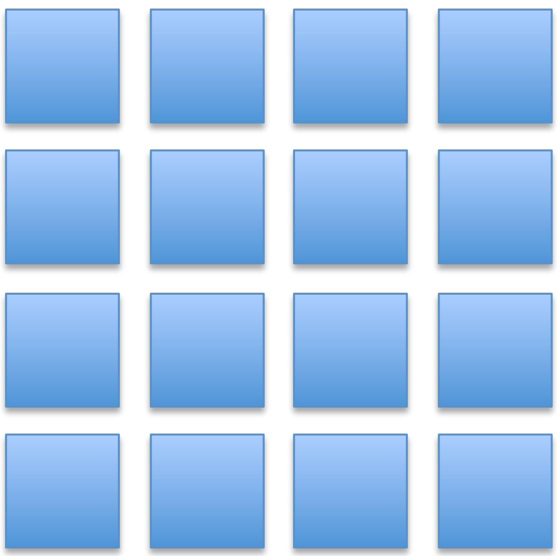
Parallel Implementation

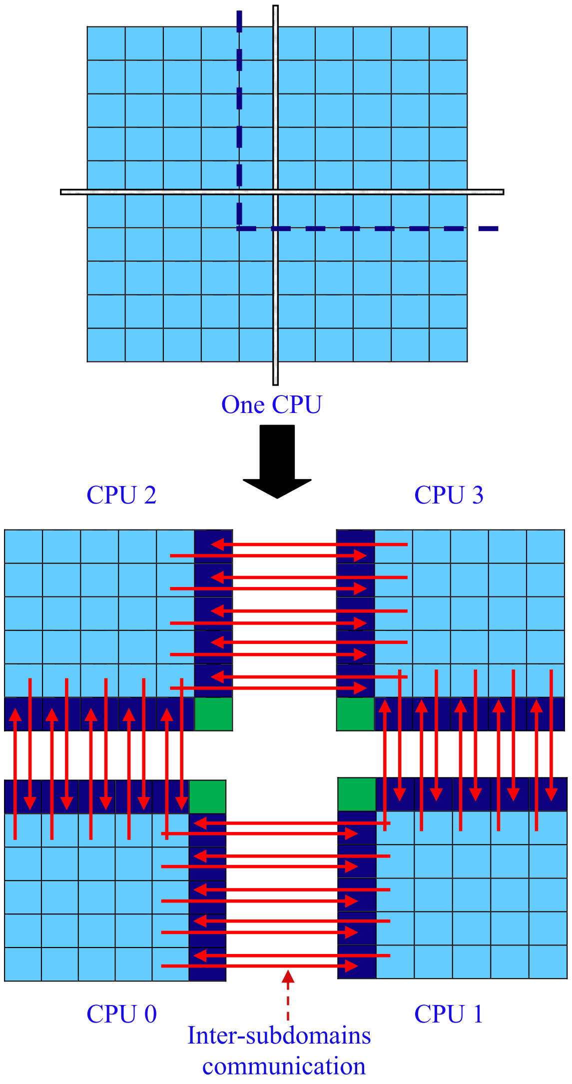
- Ghost cells



Parallel Implementation

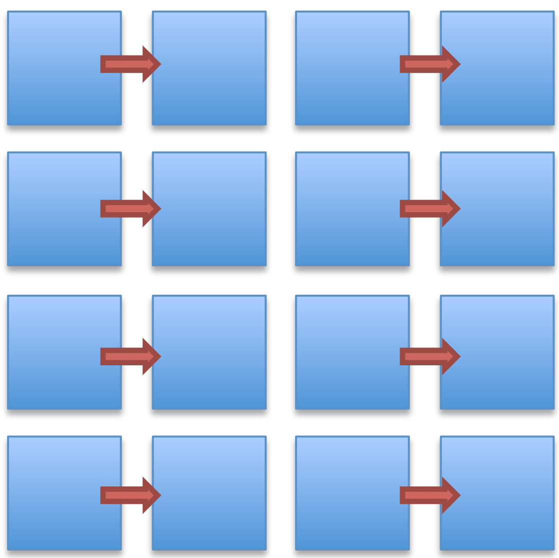
- Ghost cells

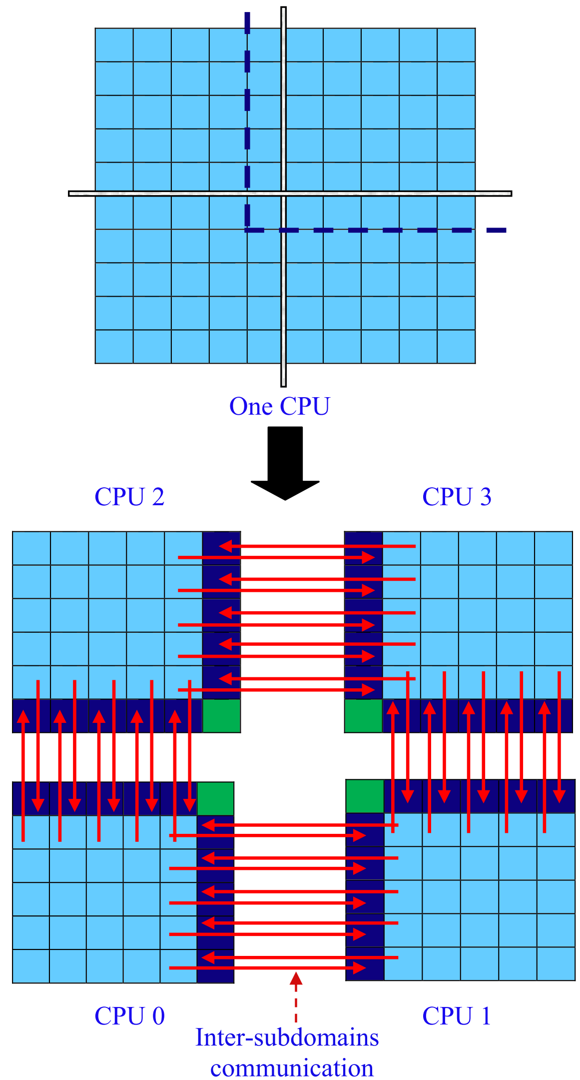




Parallel Implementation

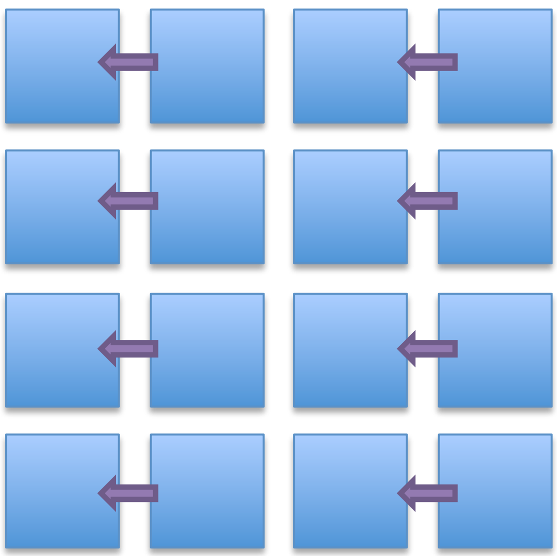
- Ghost cells

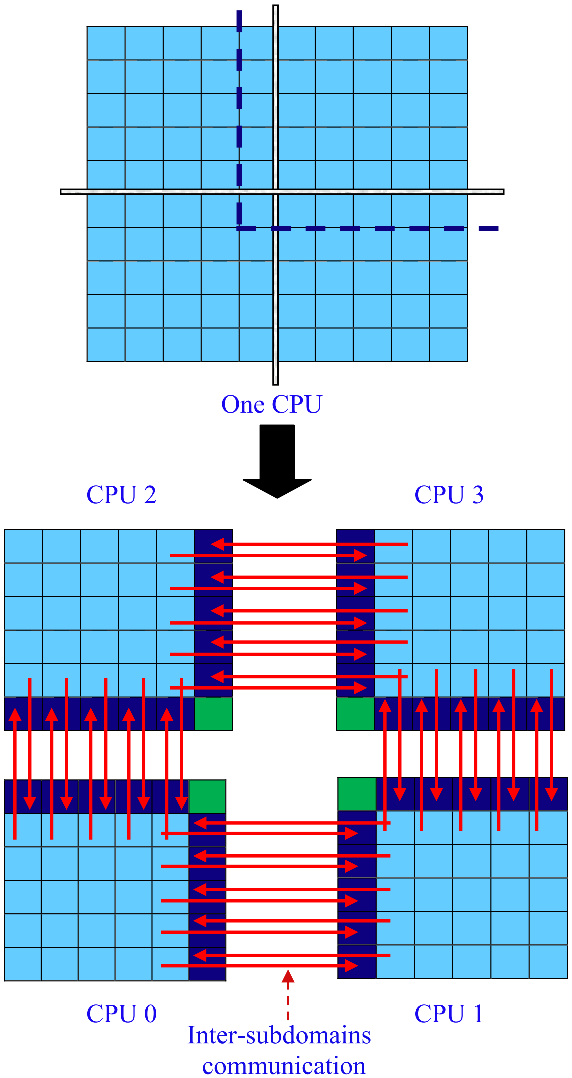




Parallel Implementation

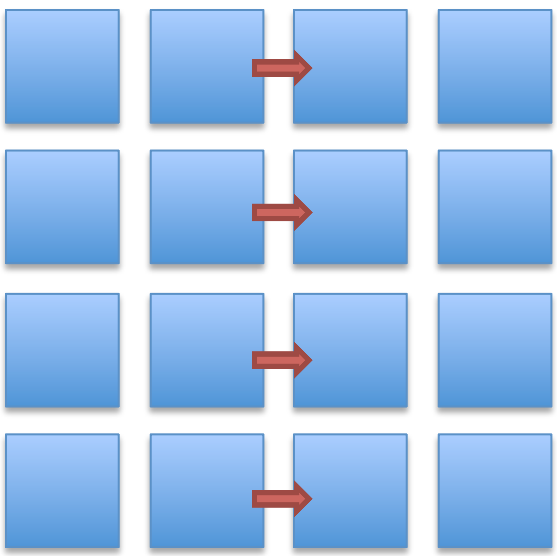
- Ghost cells

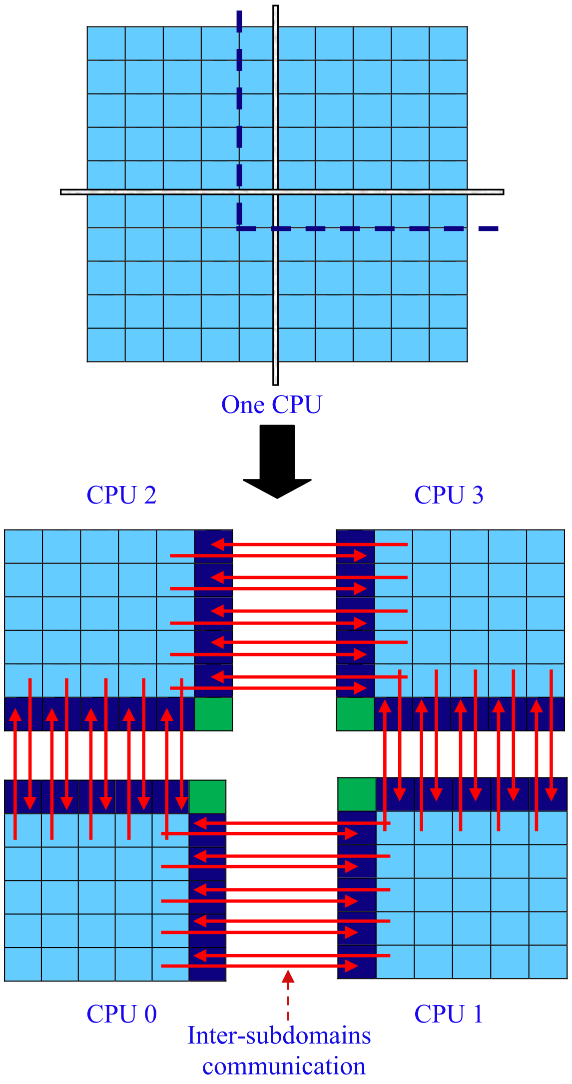




Parallel Implementation

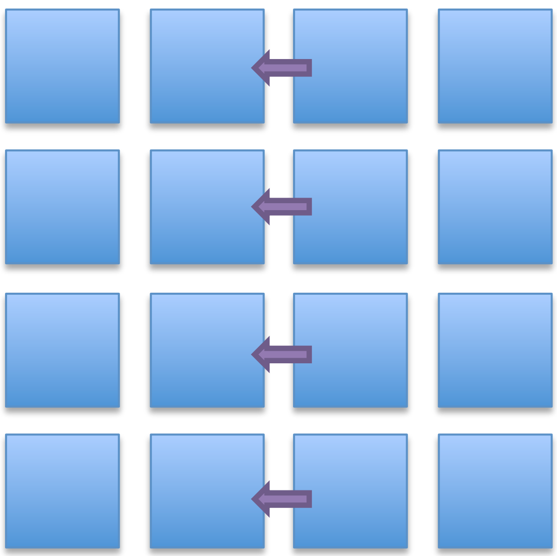
- Ghost cells

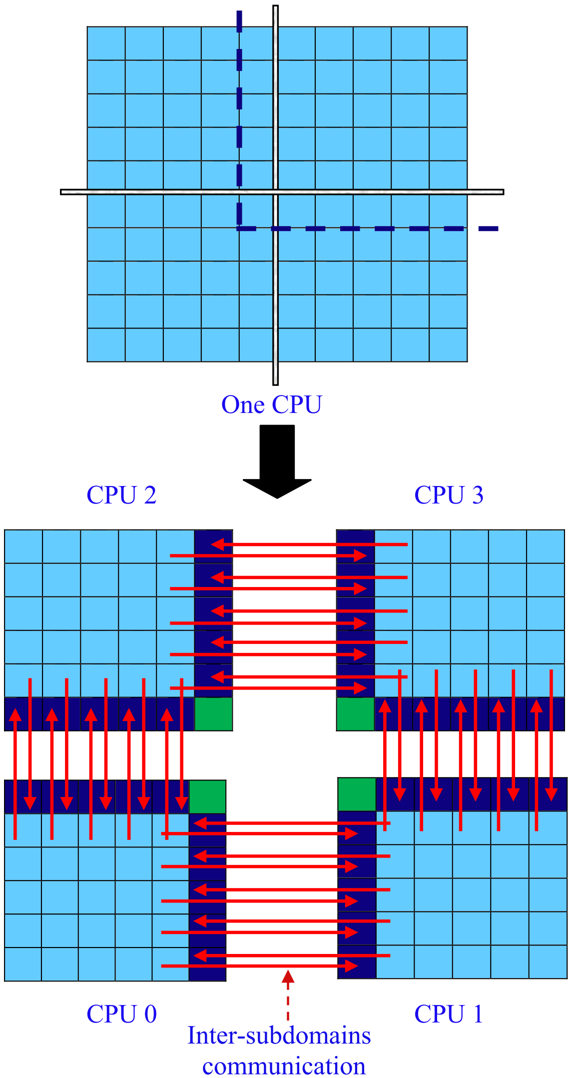




Parallel Implementation

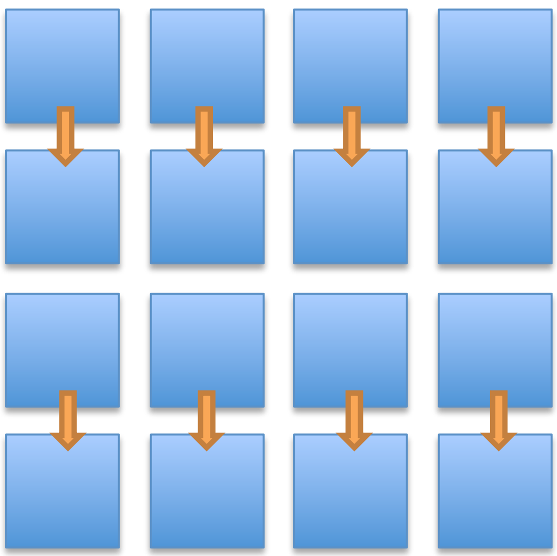
- Ghost cells

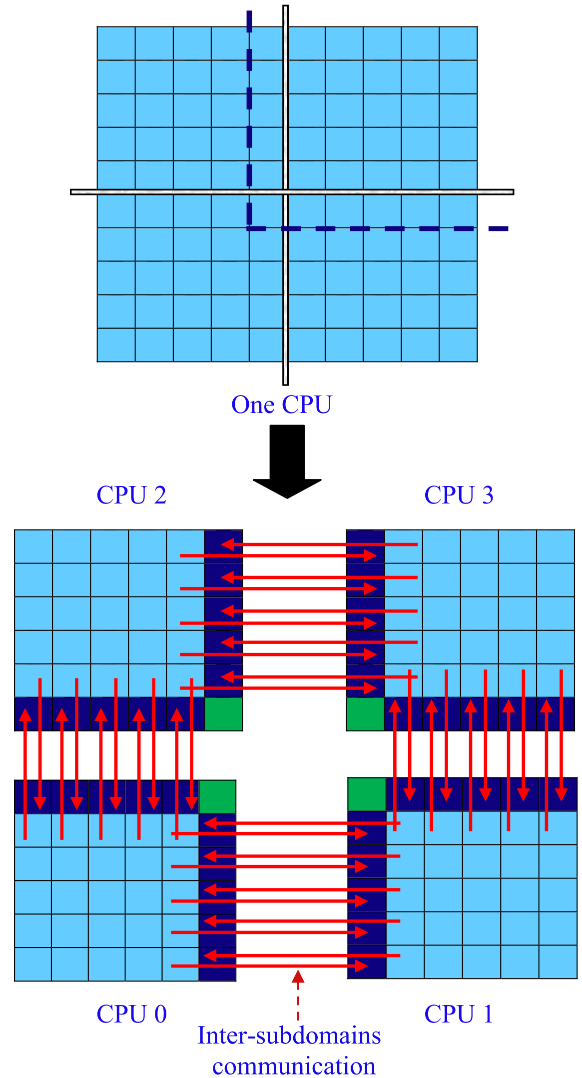




Parallel Implementation

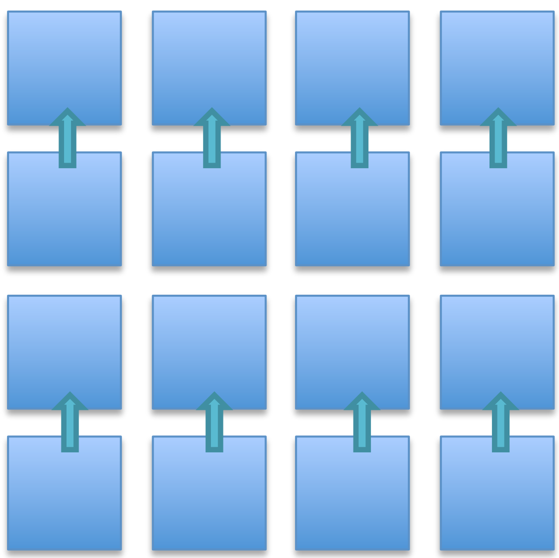
- Ghost cells

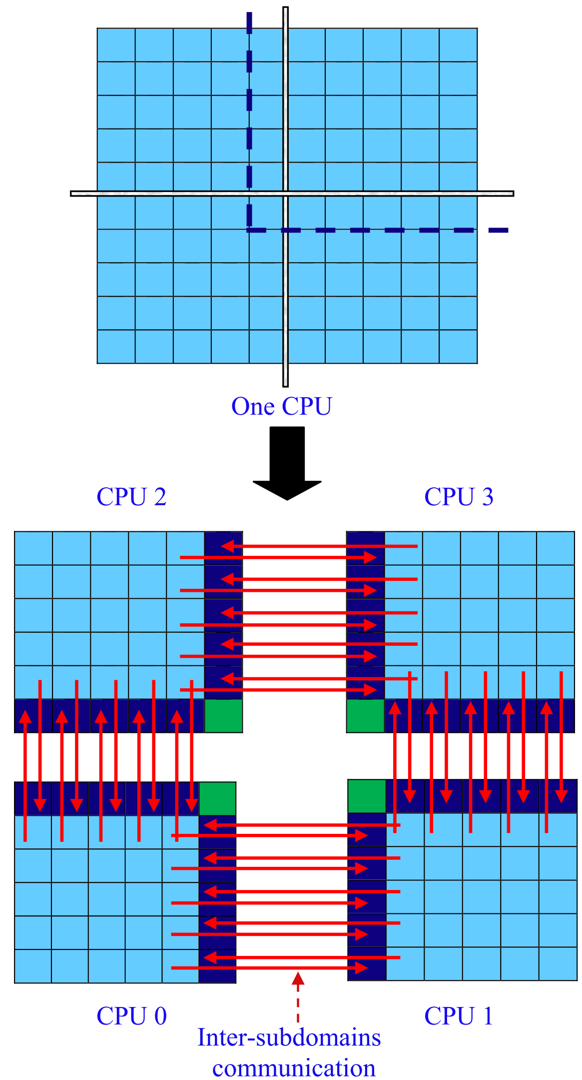




Parallel Implementation

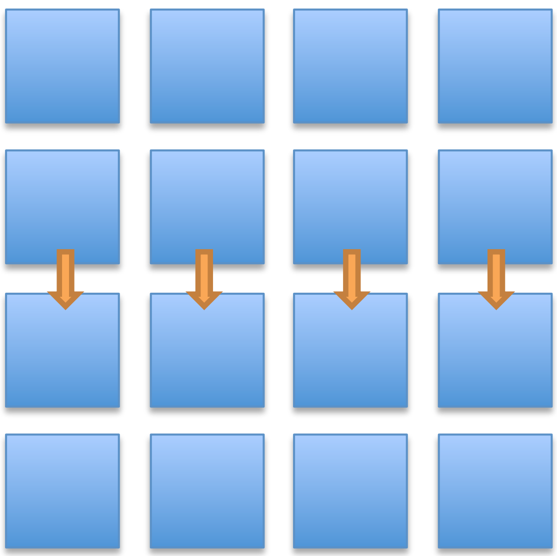
- Ghost cells

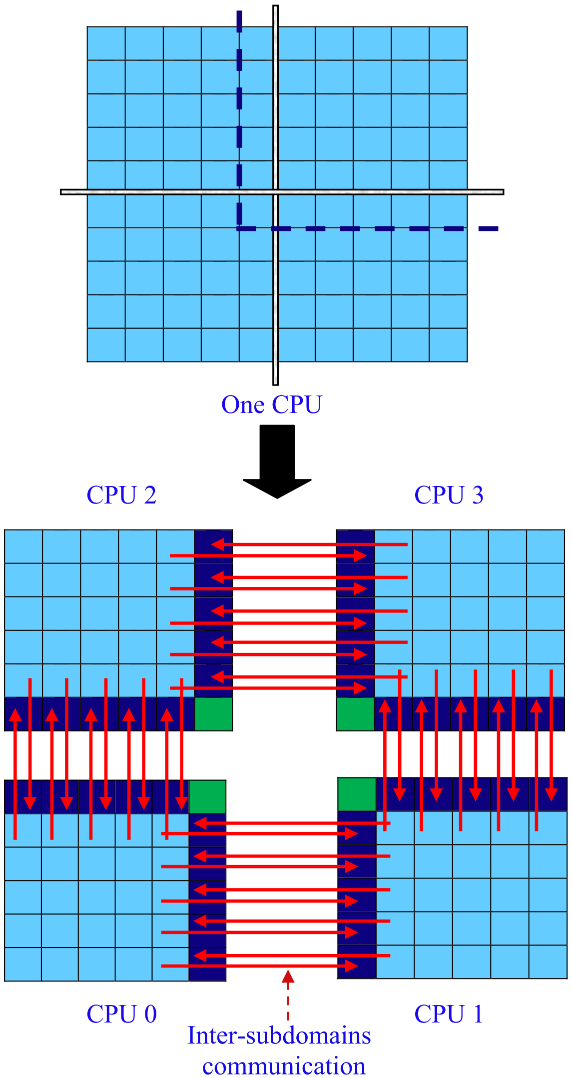




Parallel Implementation

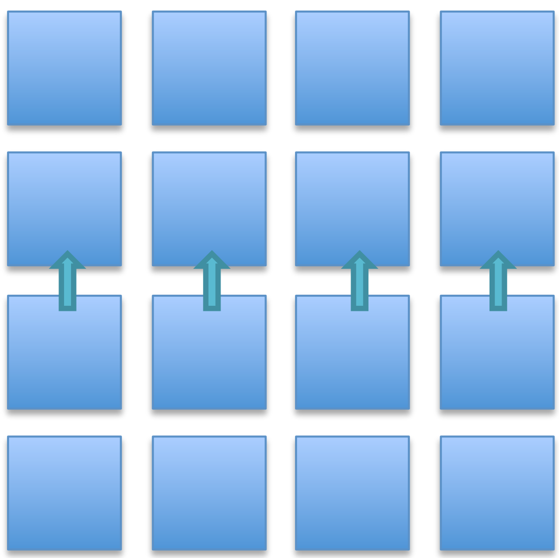
- Ghost cells

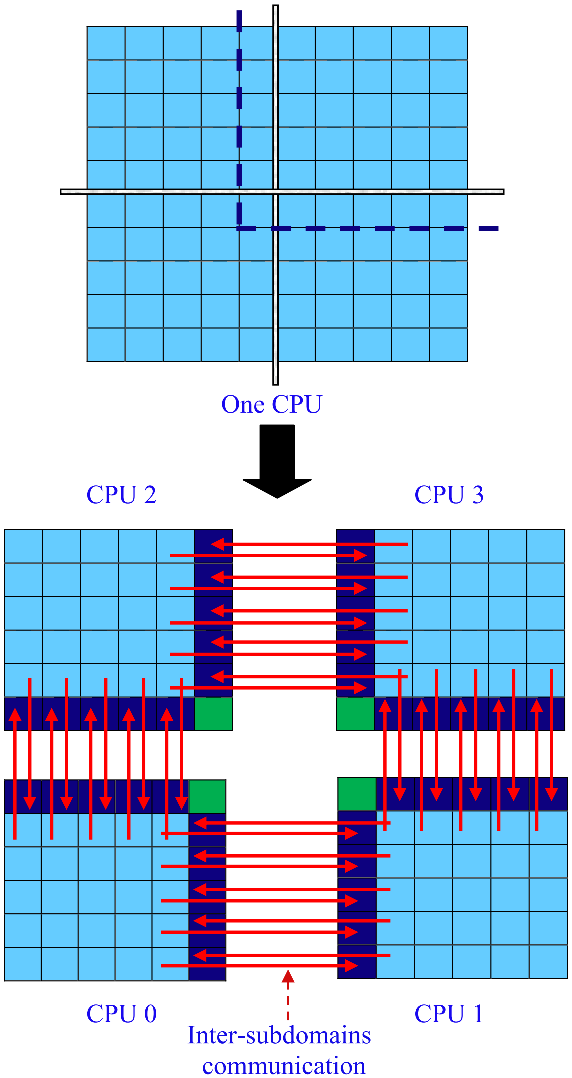




Parallel Implementation

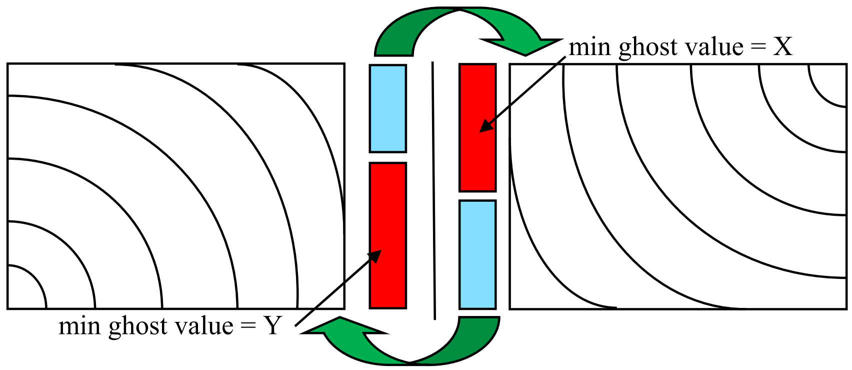
- Ghost cells



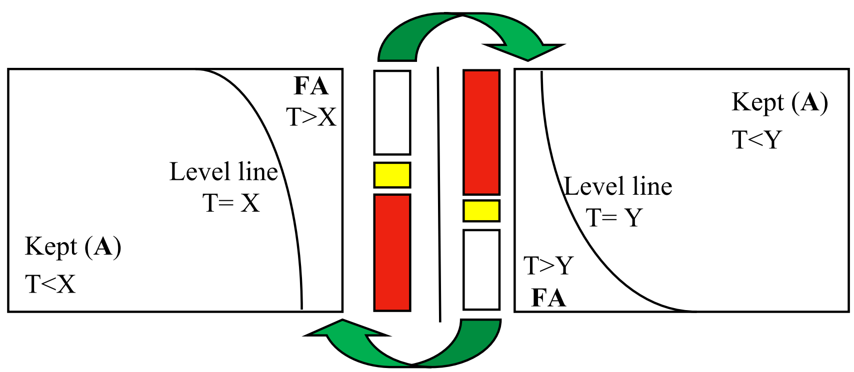


Parallel Implementation

- Fast Sweep



(a) After Local FM



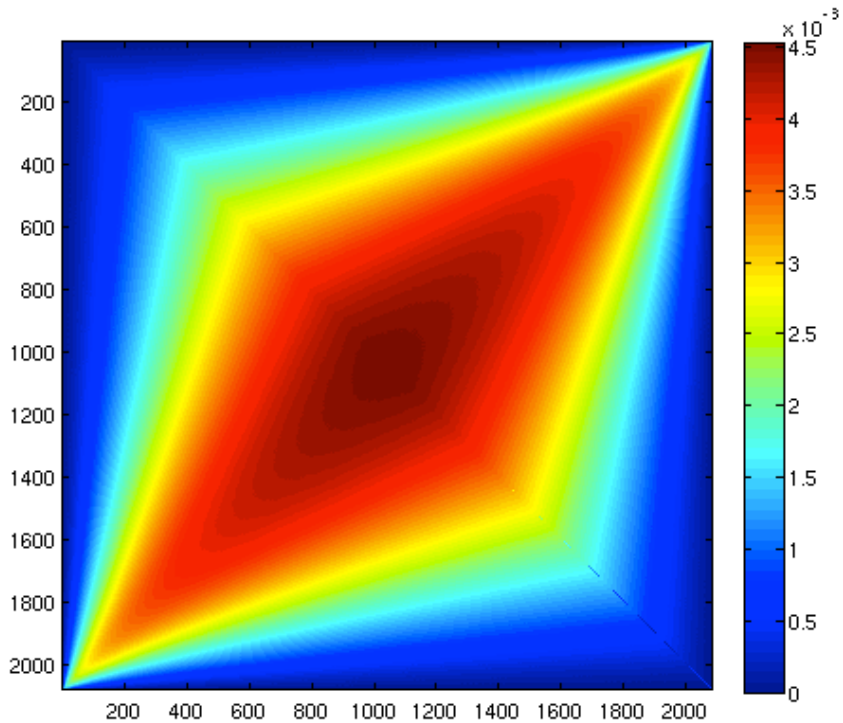
(b) After Ghost Update

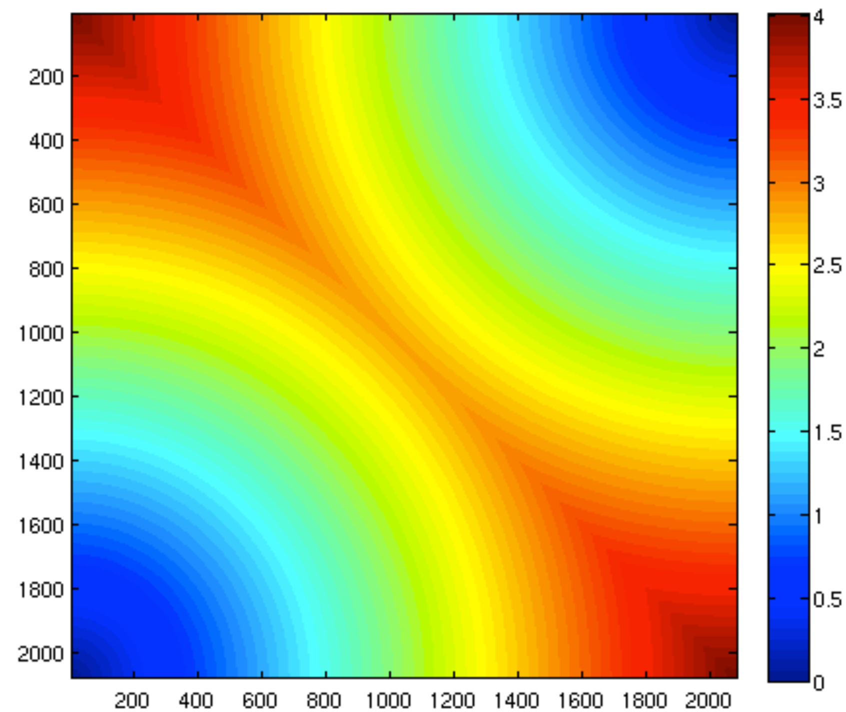
Parallel Implementation

- Iterations $nit = n + m + 1$
- Complexity $$C(N,n) = (2n+1)\frac{N^{\alpha}}{n^{\alpha}}$$
- Scability $$sc \approx \frac{n^{2,35}}{2}$$

Parallel Implementation

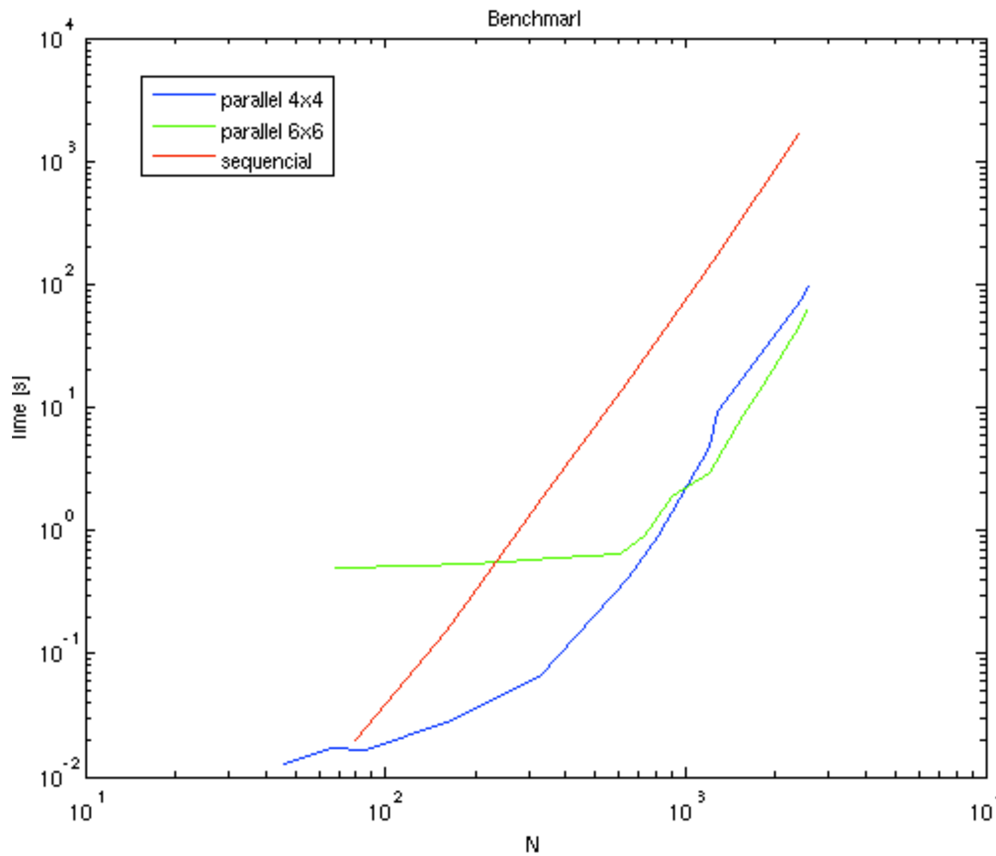
- Results





Parallel Implementation

- Comparaison



Conclusion

- Good Scability
- Unequal load of the processes

Questions