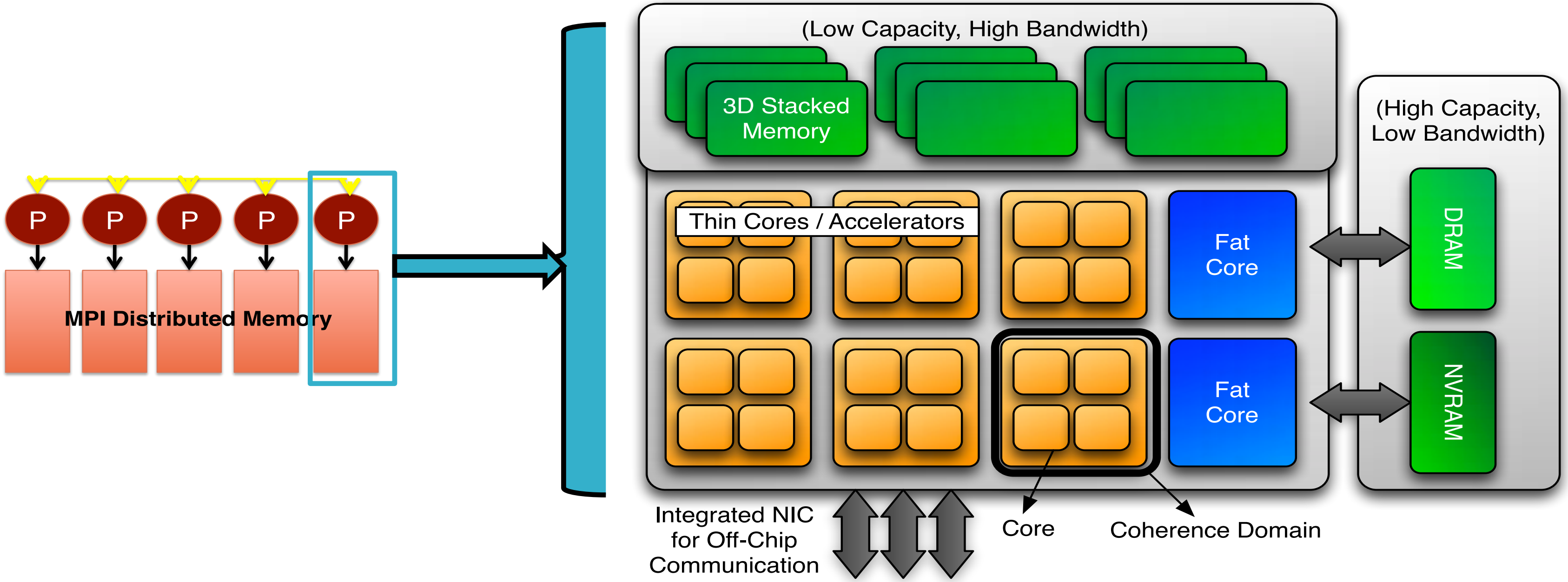
Designing a New Poisson Solver for Exascale Architectures

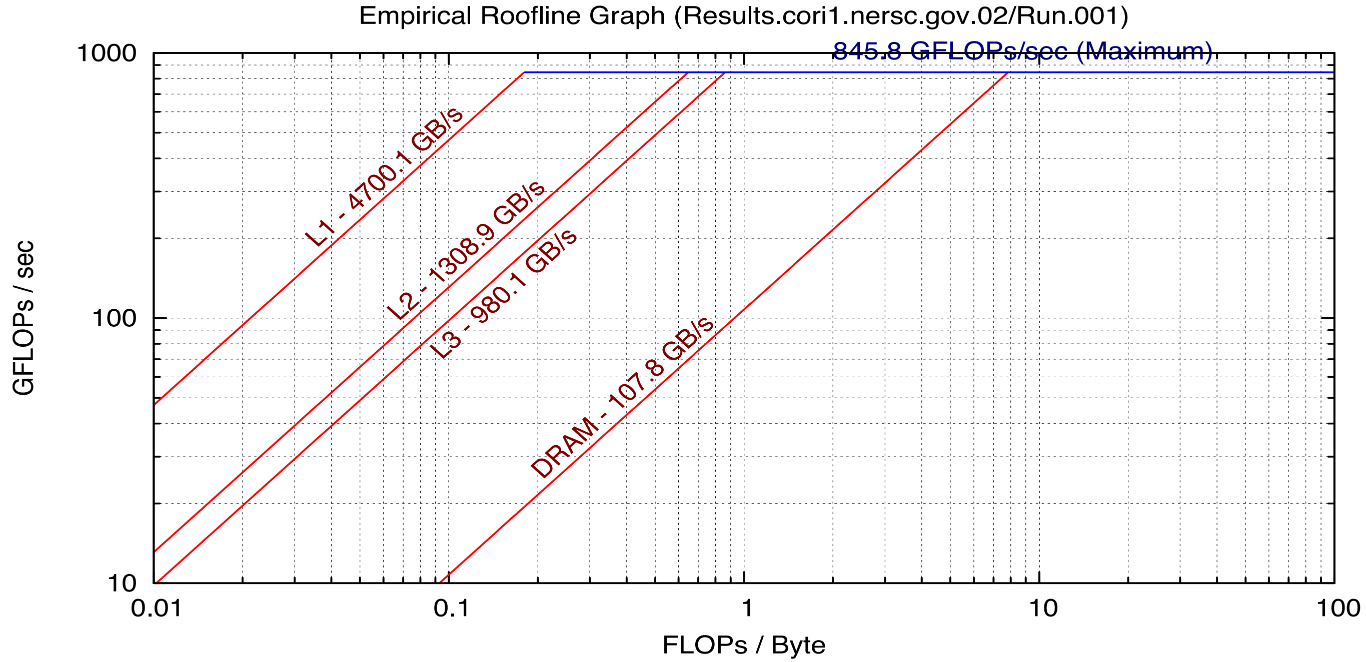
Phillip Colella
Computational Research Division

Power Constraints Lead to New HPC Architectures



- Clock cycle remains fixed (end of Dennard scaling). More FP capacity comes from more cores (Moore's law is still in operation).
- Relative size of the memory decreases. Memory architecture becomes more complex, in ways that can't be hidden from / ignored by the applications developer.
- Flops are overprovisioned relative to data motion / data storage.

Roofline Model (Williams, et al. 2009)



Arithmetic Intensity (AI) – number of flops / byte moved. Property of algorithm (including working set size).

Flops / byte below a threshold – computation is only done as fast as data can get between the arithmetic unit and memory.

Roofline gives upper bounds.

Trends: Horizontal line moving up, slanted lines remaining more or less fixed

Poisson's Equation

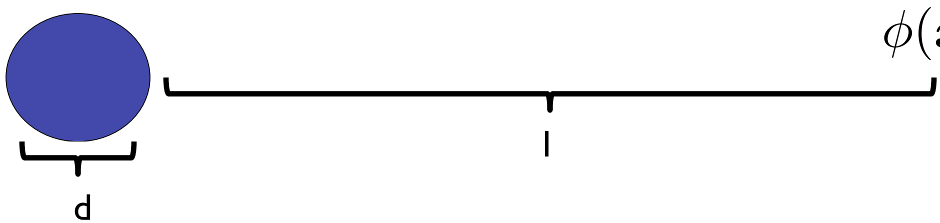
Can view it as solving a PDE, or computing the convolution with a Green's function.

$$\Delta\phi = f \Leftrightarrow \phi(\boldsymbol{x}) = (G * f)(\boldsymbol{x}) \equiv \int G(\boldsymbol{x}-\boldsymbol{y})f(\boldsymbol{y})\,d\boldsymbol{y}\ ,\ G(\boldsymbol{z}) = -\frac{1}{4\pi|\boldsymbol{z}|}.$$

Naïve implementation of convolution leads to O(N^2) calculation.

Local regularity: the field induced by a localized charged is smooth away from the support of the charge.

$supp(f)$



$\phi(\boldsymbol{x})$ $\quad \nabla^{\boldsymbol{k}}\phi = ||f||_\infty O\left(\frac{d^3}{l^{k_1+k_2+k_3+1}}\right)$

Exploiting local regularity leads to fast O(N log N) methods: Multigrid, Fast Multipole, Tree.

.

Structured-Grid Discretizations for Poisson

Mehrstellen Discretizations of Laplacian:

$$(\Delta^h \phi^h)_{\boldsymbol{g}} = \sum_{\boldsymbol{s} \in [-s,s]^3} a_{\boldsymbol{s}} \phi^h_{\boldsymbol{g}+\boldsymbol{s}}$$

$$\tau^h(\phi) = C_2 h^2 \Delta(\Delta\phi) + \sum_{q'=2}^{Q/2-1} h^{2q'} \mathcal{L}^{2q'}(\Delta\phi) + h^Q L^{Q+2}(\phi)$$

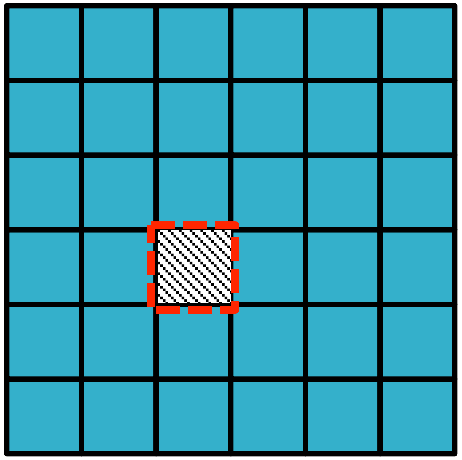
$$s = \left\lfloor \frac{Q}{4} \right\rfloor \qquad s = 1 \Rightarrow Q = 6 \ , \ s = 2 \Rightarrow Q = 10$$

Q = order of accuracy on harmonic functions. High-order accuracy of general solutions recovered by modifying the right-hand side.

$$(G^h * f^h) = (\Delta^h)^{-1}(f^h) \ , \ (G^h * f^h)[\boldsymbol{g}] \equiv \sum_{\boldsymbol{g}' \in \mathbb{Z}^3} h^3 G^h[\boldsymbol{g} - \boldsymbol{g}'] f[\boldsymbol{g}']^h$$

$$G^h[\boldsymbol{g}] = h^{-1} G^{h=1}[\boldsymbol{g}]$$

Multigrid in parallel



Domain decomposition into patches (so each step
Is actually a loop over patches):
Iterate to convergence:
Point relaxation

$$\phi \mathrel{+}= \lambda(f - \Delta^h \phi) \text{ (p times)}$$

Solve (relax) coarsened problem

$$R^C = \Delta^{2h}(Av(\phi)) + Av(f - \Delta^h \phi)$$

$$\text{solve } \Delta^{2h}\phi^C = R^C$$

Interpolate correction

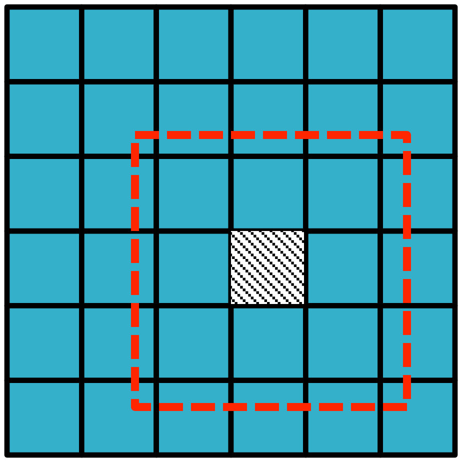
$$\phi \mathrel{+}= \mathcal{I}(\phi^C - Av(\phi))$$

Point relaxation

$$\phi \mathrel{+}= \lambda(f - \Delta^h \phi) \text{ (p times)}$$

Data moved / unknown (bytes) = # iterations (10) x 8 x (3 + 1 + 1 + 3) = 640. Every step requires moving data to and from main memory.
Flops = # iterations(10) x ((size of stencil)(7-27) x p + 1 + 8 + (size of stencil) x p) = 300-1200
AI ~ ½ - 2 .
Can we change the algorithm to better exploit local regularity to reduce data motion?

Method of Local Corrections



Domain decomposition into patches:
Compute local solution on larger, overlapping boxes

$$\phi^{patch} = G^h * f^{patch}$$

Compute coarsened solution

$$\phi^C = G^{4h} * \Big(\sum_{patches} \Delta^{4h}(\mathcal{S}(\phi^{patch}) \Big)$$

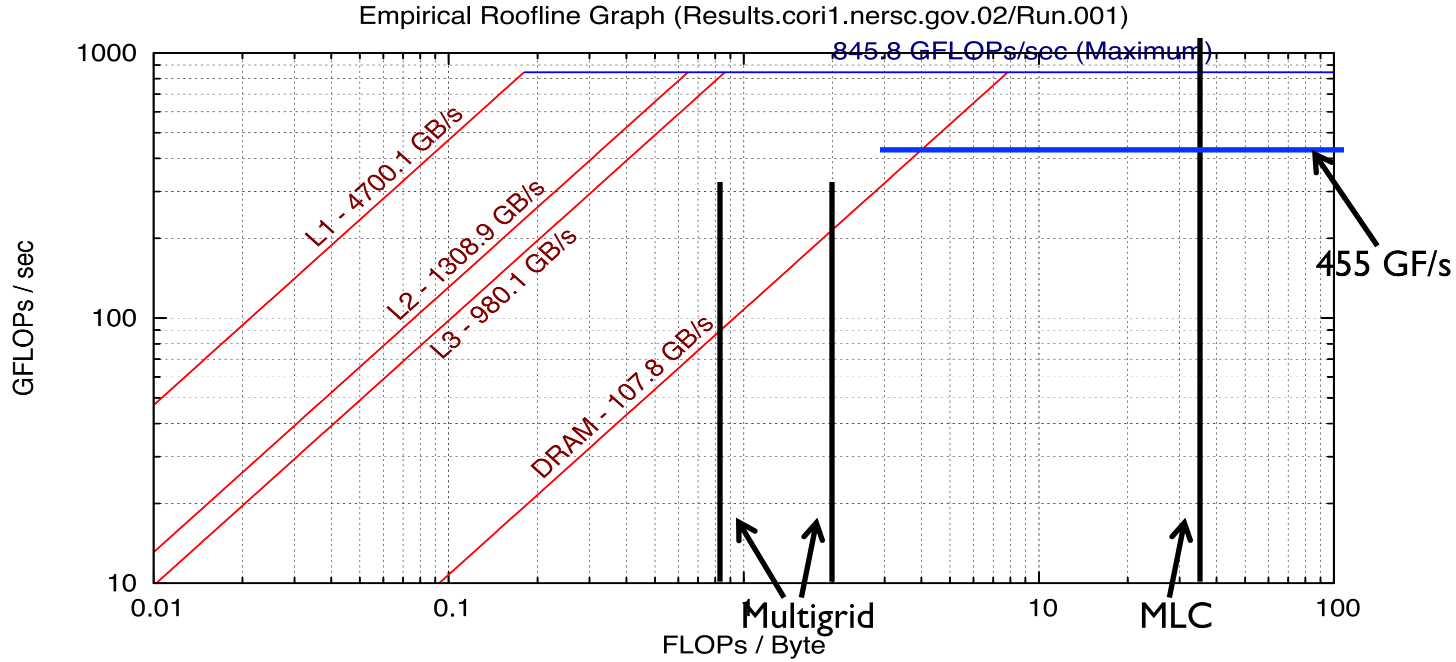
Compute boundary conditions on each patch as the sum of overlapping patches, plus interpolated correction

$$\phi^{patch} = \phi^{local} + \mathcal{I}(\phi^C - \phi^{local})$$

and solve Dirichlet problems.
Done! (no iteration)

Data moved / unknown (bytes) = ~150 bytes. Only storing results from initial calculation on Coarsened grid, boundaries of patches.
Flops / unknown = 5000, mostly in the initial local convolutions on enlarged patches.
AI ~ 33.

Multigrid vs. MLC on Roofline



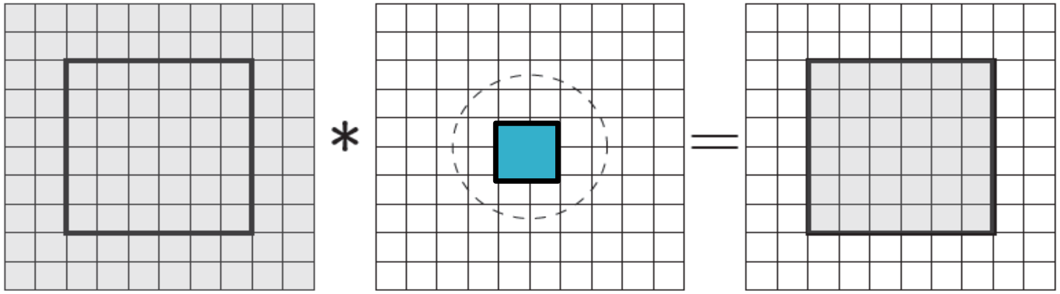
Multigrid: Ranges from 7-point to 27 point stencils.
455 Gflops/sec limit on MLC due to inability to fully utilize FMA.

Hockney's Method for Discrete Convolutions

Discrete convolutions diagonalized by discrete Fourier transforms (Hockney, 1970) .

$$\sum_{\boldsymbol{i} \in \mathbb{Z}^D} f(\boldsymbol{i} - \boldsymbol{j}) g(\boldsymbol{j}) = \sum_{\boldsymbol{j} \in B''} f(\boldsymbol{i} - \boldsymbol{j}) g(\boldsymbol{j}) = \mathcal{F}^{-1}(\mathcal{F}(f)\mathcal{F}(g))_{\boldsymbol{i}}$$

$$\text{for } \boldsymbol{i} \in B \text{ , } supp(f) \subseteq B'$$



We can increase the width of the padding so that the size of B'' leads to efficient FFTs. Convolution is now N log(N).

MLC Error Analysis

Asymptotic error estimate is given by

$$\phi^{MLC} - \phi = O(h^q) + ||f||_\infty O(\alpha N)^{-Q}$$

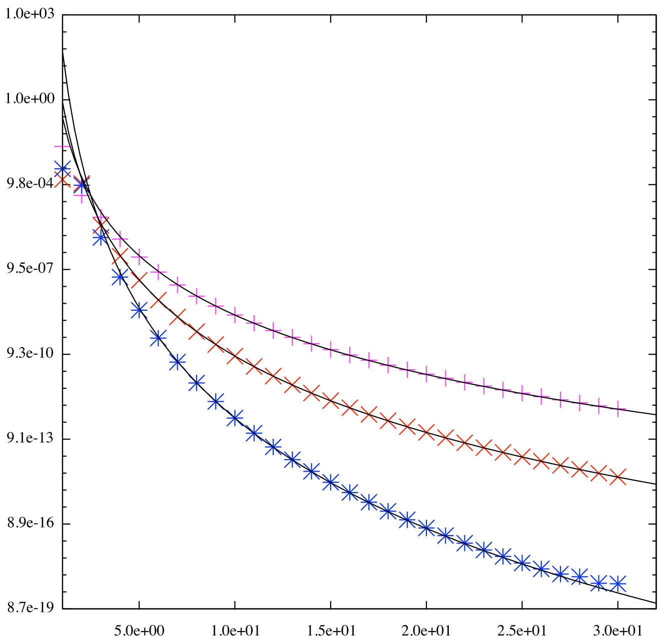
N = diameter in grid points of the source patch, αN = diameter of the destination patch. The first term is the error from the local convolutions, the second term is the error due to coarse-grid representation of the global coupling.

h > $h_{threshold}$: solution error goes down under grid refinement.
h < $h_{threshold}$: solution error no longer goes down under grid refinement (localization error).

Potential lever arms and performance tradeoffs in choosing α, N, Q .

Discrete Potential Theory of Mehrstellen Operators

Our error estimate is a direct consequence of the rapid decay of the truncation error of the operator as a function of max-norm distance from the charge. The decay rate increases with Q.



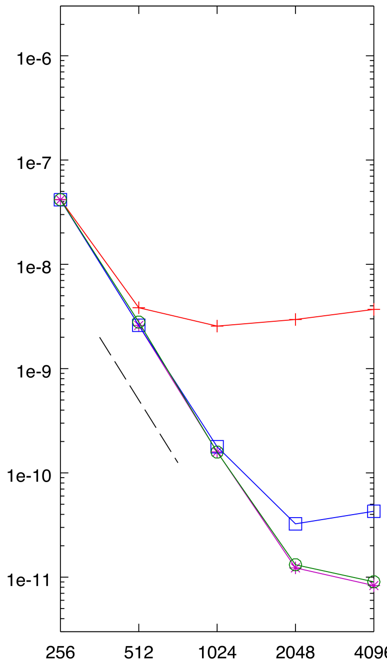
Plot of

$$\max_{||\boldsymbol{i}||_\infty = i} |(\Delta^{h=1} G)_{\boldsymbol{i}}|$$

for Q=4 (magenta) Q =6 (red) and Q=10 (blue). Solid lines are power-law fits ~$i^{(Q+3)}$. (Exact answer is 0).

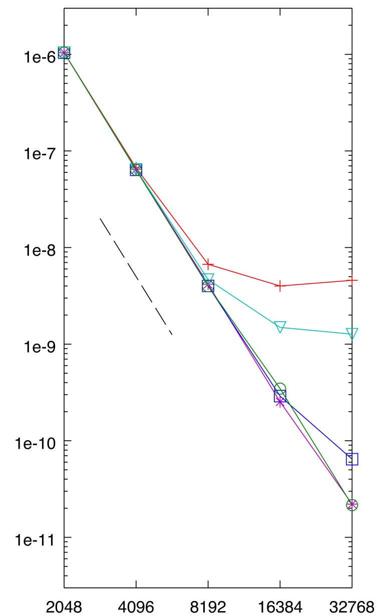
Results: Accuracy

Errors are all max norm errors scaled by the max norm of the solution.



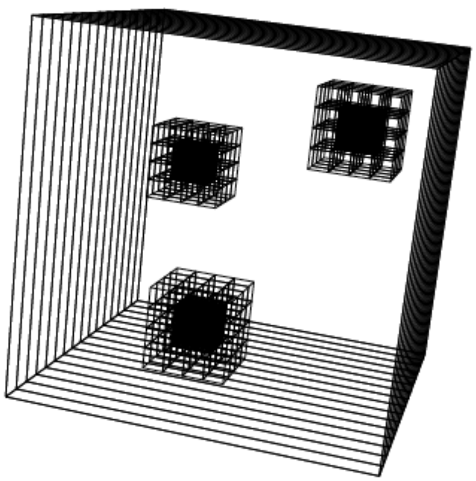
Uniform-grid test case:

- +: $Q=6,\ \alpha=3.25,\ N=32$
- □: $Q=6,\ \alpha=3.25,\ N=64$
- ✳: $Q=10,\ \alpha=3.25,\ N=32$
- ○: $Q=10,\ \alpha=3.25,\ N=64$



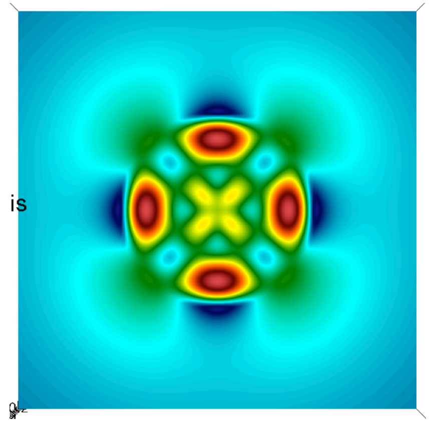
Adaptive-grid test case:

- +: $Q=6,\ \alpha=3.25,\ N=32$
- ▽: $Q=6,\ \alpha=2.125,\ N=64$
- □: $Q=6,\ \alpha=3.25,\ N=64$
- ✳: $Q=10,\ \alpha=3.25,\ N=32$
- ○: $Q=10,\ \alpha=3.25,\ N=64$

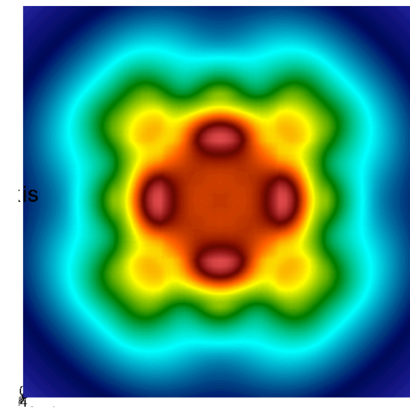


MLC: transition from local error to localization error.

Q = 6: transition from local error to smooth localization error as we refine the mesh.

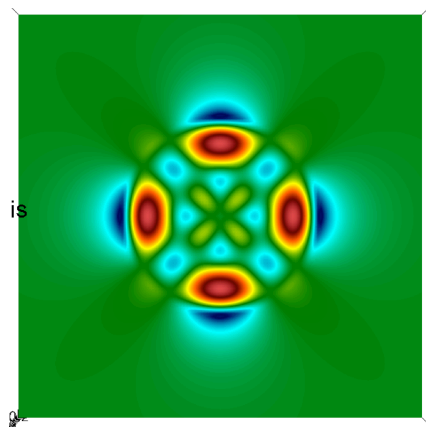


Q=6, N=512

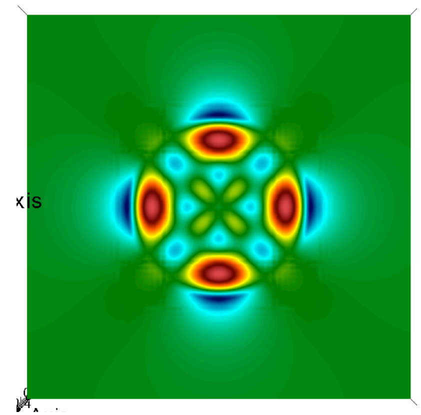


Q=6, N=1024

Q = 10: localization error is still small relative to the local error.



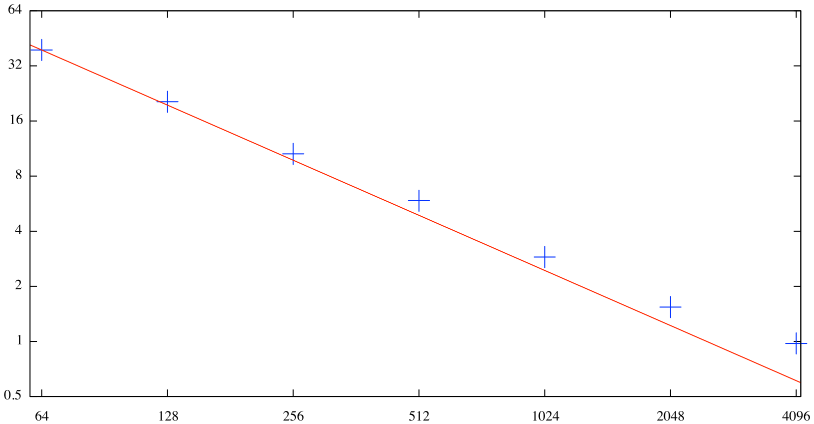
Q=10, N=512

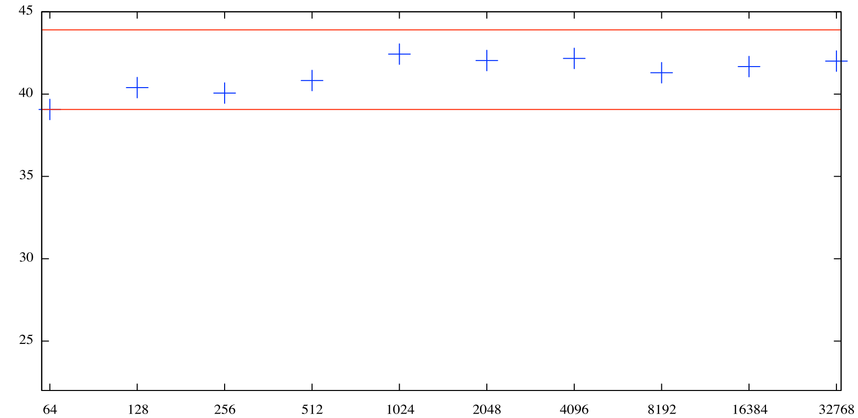


Q=10, N=1024

Performance and Scaling of MLC on NERSC Cori I

Numerical parameters: $Q = 6$, $\alpha = 3.25$, $N = 32$. Plots are of wall-clock time to solution (in seconds) vs. number of cores. Red lines represent perfect scaling (or 10% slower).





Strong scaling: Fixed problem size with 10^9 grid points, adaptive distribution (0.2% of domain refined at finest level), using 64 – 4K cores. Greater than 60% strong scaling efficiency over that range.
Time to solution 39.1 -> .97 seconds.

Replication weak scaling: 10^9 grid point adaptive base case, replicated to obtain larger problems, computed on 64 – 32K cores. Solution error is independent of scale (~7 x 10^{-9}).

- 92% weak scaling efficiency. Time to solution 39.1 – 42.4 seconds.
- Largest calculation has 5.1 x 10^{11} unknowns, with an equivalent uniform-grid resolution of $(64K)^3$ = 2.8 x 10^{14} unknowns.

COMPUTATIONAL RESEARCH DIVISION

BERKELEY LAB

Performance Comparison

Cori I (Haswell) 32 cores / node, 8 nodes.

Multigrid: 10^9 unknowns (1024^3 grid) using HPGMG (non-adaptive) benchmark, 10 v-cycles: 9 seconds, 17 Gflops / node (RLB = 200 Gflops).

MLC: 10^9 unknowns, both uniform and adaptive cases: 8 seconds, 70 Gflops / node. Hockney kernel: 3.64 seconds, 140 Gflops / node (RLB = 455 Gflops).

MLC isn't coming close to the roofline (nor is Multigrid, for that matter). Why not ?

- In MLC case, the non-unit stride access to set up SIMD is one of the problems. A possible cure is to use split representation in FFT ({real array, imaginary array}, rather than array of {real,imaginary}). But FFTW doesn't do that (not really – the split API is supported, but it copies the data into the non-split format).

- Data choreography needs more squeezing.

Algorithmic Details

- Use of Mehrstellen discretizations: 3x3x3 stencil for Q=6; 5x5x5 stencil for Q=10. The use of such compact stencils minimizes the size of the local convolutions for a given degree of overlap.

- Reduce the size of the local convolution by representing the solution in an outer annulus $\alpha_0 R \leq ||\boldsymbol{i}h||_\infty \leq \alpha R$ in terms of a low order Legendre expansion of the charge. Only the expansion coefficients need to be computed, and precomputed convolutions of Legendre polynomials with the Green's functions read from a table.

- Computational kernel is discrete convolution on modest-sized patches (input length 33, output length 110-140). Initially, one patch / core. Multiple threads working on a single patch decreases the extent to which L3 must be shared – Hockney fits into L3, with only the minimum traffic to DRAM.

Future Work

- Making it software (almost ready for a 1.0 release as part of the Chombo distribution).
- We are still exploring the performance / accuracy tradeoff space. Patch size (N,alpha), overlap (alpha), harmonic order of accuracy (Q).

$$\phi^{MLC} - \phi = O(h^q) + ||f||_\infty O(\alpha N)^{-Q}$$

- We need to take a deep dive into FFT - substantial amount of difficult, platform-dependent work to get high performance Hockney kernels. This kind of work can be automated (symbolic transformations, code generation).
- Fast local convolutions suggest alternative approach to solving constant-coefficient PDE. Convolution methods for Maxwell ? Fast evaluation for high-order stencils ?

Final Comments

“We make very limited claims to novelty for [the methods] presented here. The ideas involved are quite standard and, indeed, old-fashioned.”

Hockney, 1970; Mayo, 1984 (finite difference localization for boundary integrals); Anderson, 1986 (MLC for particles).

Changing the math can have a large lever arm.

Collaborators:
Scott Baden, Greg Balls, Christos Kavouklis, Peter McCorquodale, Brian Van Straalen.