A massively parallel butterfly algorithm for applying Fourier integral operators

Jack Poulson, Lexing Ying

Center for Numerical Analysis, Institute for Computational Engineering & Sciences, University of Texas at Austin

Introduction

The butterfly algorithm[1] approximates

$$\int_{\mathbb{R}^d} e^{i\Phi(x,k)} f(k) dk,$$

where $\Phi(x,k)$ is the *phase function* and is linear in its second argument. When the sources and potentials are all located on a Cartesian grid over d-dimensional unit cubes with spacing $1/N$, the naïve complexity of $O(N^{2d})$ is reduced to the near-optimal $O(N^d \log_2(N))$.

Zeroth order picture:

- Start with N^d global low-rank potentials from localizes sources
- End with N^d local low-rank potentials from global sources

The majority of the computation lies in the recursions for the coefficients of the low-rank potentials. During the first half of the algorithm, one uses the **frequency weight recursion**:

$$\delta_t^{AB} = e^{-iN\Phi(x_0(A),p_t^B)} \sum_c \sum_{t'} L_t^B(p_{t'}^{B_c}) e^{iN\Phi(x_0(A),p_{t'}^{B_c})} \delta_{t'}^{A_p B_c},$$

while the second half uses the **spatial weight recursion**:

$$\delta_t^{AB} = \sum_c e^{iN\Phi(x_t^A,p_0(B_c))} \sum_{t'} L_{t'}^{A_p}(x_t^A) e^{-iN\Phi(x_{t'}^{A_p},p_0(B_c))} \delta_{t'}^{A_p B_c}$$

Parallelization

Requirements:

- Power of two number of processes, say 2^P
- $2^P \leq N^d$

Main ideas:

- Use partitions to spread interaction pairs across all processes
- When processes collide in frequency domain, partition the spatial domain
- During collisions, weight recursions are parallelized by splitting the child summation loop
- Reduction and spatial partitioning combined with **MPI_Reducescatter** summation

Demonstration

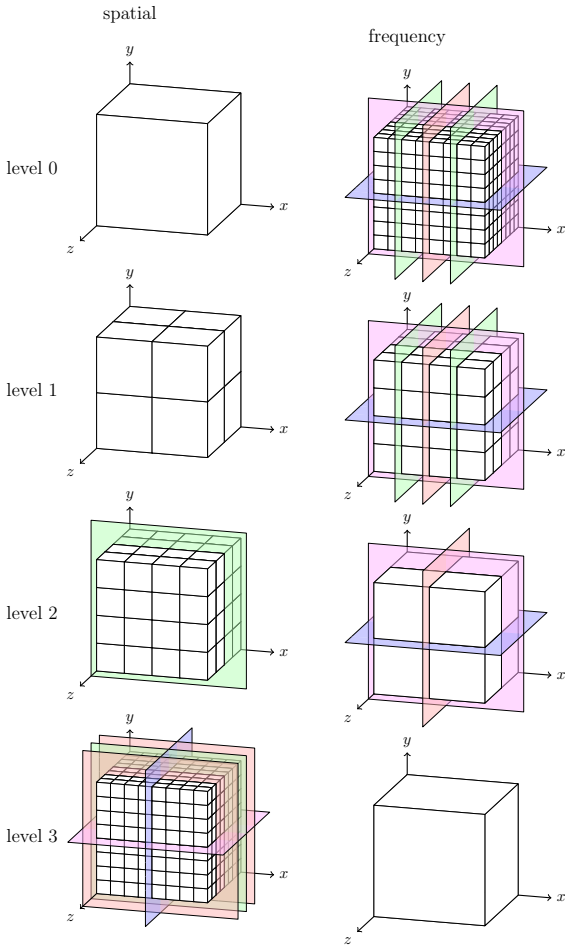

Fig. 1 A demonstration of the spatial and frequency partitions throughout the course of the butterfly algorithm in three dimensions with 2^4 processes and $N = 2^3$. The red, blue, magenta, and green planes respectively partition the nodes based upon the first, second, third, and fourth bits of the binary representation of their ranks.

Results

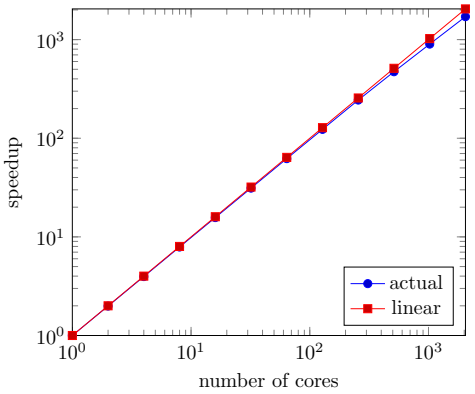

Fig. 2 *Strong scaling results of a 2d generalized Radon transform on up to 2048 cores of a Blue Gene/P. The corresponding phase function was*

$$\Phi(x,k) = x \cdot k + \sqrt{c_1^2(x)k_1^2 + c_2^2(x)k_2^2},$$

where $c_1(x) = (2 + \sin(2\pi x_1)\sin(2\pi x_2))/3$ *and* $c_2(x) = (2 + \cos(2\pi x_1)\cos(2\pi x_2))/3$.

Conclusions and future directions

An extremely scalable method for parallelizing the butterfly algorithm for Fourier integral operators with unit amplitude functions has been demonstrated. A natural next step is to investigate methods for handling a wider class of amplitude functions. Another interesting direction is to investigate using the method as a replacement for the non-scalable FFT on massively parallel architectures[2].

Obtaining the source code

The *butterflyfio* project and source code are hosted on Google Code at http://code.google.com/p/butterflyfio

References

[1] E. Candès, L. Demanet, and L. Ying. A fast butterfly algorithm for the computation of Fourier integral operators. SIAM MMS, 2009.

[2] A. Edelman, P. McCorquodale, and S. Toledo. The future Fast Fourier Transform. SIAM SISC, 1997.