

Spring 2019: Advanced Topics in Numerical Analysis: High Performance Computing

Example projects

- **Parallel multigrid for the Laplace equation.** Multigrid is an optimal complexity solver for linear systems arising from elliptic partial differential equations or from certain networks. We will present the basics of Multigrid, which heavily builds on the Jacobi and Gauss-Seidel methods we have already used in class, in one of our lectures. A project could focus on the implementation of parallel multigrid using finite-difference approximation on a square or a cube either using MPI or CUDA.¹
- **Total variation image denoising.** Implement a total variation image denoising algorithm.² Total variation regularization is a method to remove noise from images while maintaining sharp edges in the image (i.e., not blurring them, as other filters would do). Over the last decade, first-order methods using simple ideas from convex analysis (you don't need to know these for this project) for these problems have proven to be useful. These algorithms map nicely on CPUs and GPUs as they do not require the solution of linear systems. You could implement the algorithm first on a CPU and then on a GPU.
- **Adaptive finite volume solution of the wave equation.** Parallel adaptive solution of the wave equation using the finite volume method, using the **p4est**³ library for adaptivity. The library supports parallel adaptive mesh refinement and includes an example for the adaptive solution of the advection equation already. We are well familiar with that library, which was developed by collaborators, and can thus provide help.
- **Fast Fourier Transform (FFT).** The FFT is on the list of Top 10 numerical algorithms⁴ developed in the last century. One of its applications is to use it for solving the discrete Laplace equation (on a unit sphere or cube), possibly in parallel. Introduction material can be found easily on the web⁵. One project could be to implement the FFT algorithm using OpenMP or CUDA.
- **Conway's game of life in parallel** Conway's game of life⁶ is an example for a cellular automaton that follows simple update rules from one generation to the next. Despite their simplicity, these rules generate an impressive amount of complexity and beauty.⁷ Write an MPI-parallel (or a GPU) version of Conway's game of life. Visualization is a

¹Standard references are: Trottenberg, Oosterlee and Schuller: *Multigrid*, Academic Press (); Briggs, Henson and McCormick: *A Multigrid Tutorial*, SIAM, (2000); See also <https://people.cs.uchicago.edu/~risi/NIPS15workshop/slides/Safro.pdf>.

²<https://pdfs.semanticscholar.org/2f3c/c010e6b137542315804fa3f7ad776a5ca1ad.pdf>

³www.p4est.org

⁴<https://www.siam.org/pdf/news/637.pdf>

⁵See for instance <http://www.cs.berkeley.edu/~demmel/cs267/lecture24/lecture24.html>.

⁶See the Wikipedia page: https://en.wikipedia.org/wiki/Conways_Game_of_Life.

⁷See videos on youtube: https://www.youtube.com/results?search_query=conway+game+of+life.

bit tricky, you can either write to files and visualize externally or try to connect python to your code and use python's visualization (e.g., in pygame). Or try to google around for different solutions.

- **Parallel k-means clustering (or another clustering algorithm)** The k-means algorithm⁸ is a data mining algorithm for clustering data. Implement it either using MPI or CUDA and find several large data sets to apply it to.
- **Matrix factorizations on a GPU** Implement a matrix factorization (e.g., LU or QR) on a GPU. A quick search will show you that efficient implementations of these operations have been done before and that it is non-trivial to get good performance. Attempt an implementation and study the performance in comparison to the cuBLAS library.
- **Parallel Delaunay triangulation** This is used for constructing good quality triangle meshes from point clouds. Several algorithms have been proposed for constructing Delaunay triangulation (see <https://dl.acm.org/citation.cfm?id=1137900> and the section on related work in that paper for a summary of existing algorithms). Pick any one of the existing algorithms or propose your own scheme and implement a parallel (using either OpenMP, MPI or CUDA) algorithm for Delaunay triangulation in 2D or 3D.
- **Fluid mechanics simulation** If you took the CFD class offered by our colleague Aleks Donev last semester, you could take one of the final projects from there⁹ and port it to a parallel architecture or a GPU.
- **Parallel implementation of fast summation with Ewald sums.** Study the paper <http://dx.doi.org/10.1016/j.jcp.2010.08.026>, which describes a fast summation method for periodic Stokes potentials. Understand and implement the method in parallel.
- **Parallel partitioning of large point clouds using an adaptive octree.** In many applications, it is necessary to distribute a (very) large set of points in two or three dimensions in chunks of nearby points that roughly contain the same number of points. One way to achieve that is through the use of an adaptive octree as follows: Enclose all points in a (large enough) cube, and adaptively refine the cube until every sub-cube (also called “leaf of the octree”) contains roughly the same number of data points. Then, a uniform parallel distribution of these sub-cubes results in a roughly uniform distribution of points of the data set. Such an adaptive refinement of the space is often also the first step needed in the fast multipole method (FMM). You can use the adaptive octree library `p4est`¹⁰, which implements the required octree functions. Use a non-uniformly distributed (very) large point cloud data set (many such data sets are available on the web¹¹ to test your algorithm.

⁸https://en.wikipedia.org/wiki/K-means_clustering

⁹<https://cims.nyu.edu/~donev/Teaching/CFD/Assignments.html>

¹⁰www.p4est.org

¹¹See, for instance <http://www.pointclouds.org/news/2013/01/07/point-cloud-data-sets/>.