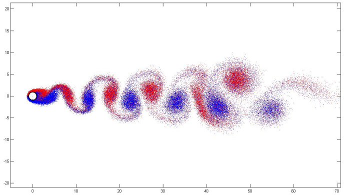
Parallel Adaptive Fast Multipole Method: application, design, and more...



Stefan Engblom

UPMARC @ TDB/IT, Uppsala University

PSCP, Uppsala, March 4, 2011

Outline

- Background: design of vertical axis wind turbines
- Discretization through vortex formulation
- Fast multipole method...
- ...with adaptivity...
- ...and done in parallel

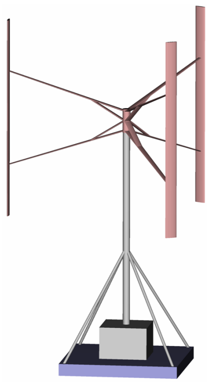
Joint work in part with **Paul Deglaire** and **Anders Goude** at the Division for Electricity and Lightning Research, Uppsala University.

Background

Pros/cons of VAWTs:

+ Generator at ground level
+ Less gravitational loads
+ No gears
+ Easier maintenance
+ Less noise
- Fatigue loads
- Start-up
- **Aerodynamics model**

YouTube: Vertical Wind 200kW (March 2010)



Vortex formulation (the *very* short version)

In 2D, let the velocity field $\mathbf{u}(z,t)$ solve the Navier-Stokes equations with BCs (using the complex number $z = x + iy$ for the space coordinate (x, y)). Introduce the *vorticity* $\omega \equiv \nabla \times \mathbf{u} \cdot \hat{k}$ and consider the two-step formulation:

$$\begin{aligned} \omega_t + \mathbf{u} \cdot \nabla \omega &= 0 && \text{(advection)}, \\ \omega_t &= \nu \Delta \omega && \text{(diffusion)}. \end{aligned}$$

-Hence; how do we obtain $\mathbf{u}$ from ω?

One can show that $\mathbf{u} = \mathbf{u}_\omega + \nabla\phi$ for some ϕ s.t. $\Delta\phi = 0$ accounting for the BCs. In turn,

$$\mathbf{u}_\omega(z,t) = \int_\Omega K(z - z')\omega(z',t)\,dz',$$

where $K = -i/(2\pi z)$ is the *Green's function* for $-\Delta$.
If the vorticity is discretized,

$$\omega(z,t) = \sum_j \delta(z - z_j)\Gamma_j,$$

with $z_j = z_j(t)$, then the velocity field is obtained from

$$\mathbf{u}_\omega(z,t) = \sum_j K(z - z_j)\Gamma_j.$$

To *advect*, evaluate the velocity field in all vorticity points z_j,

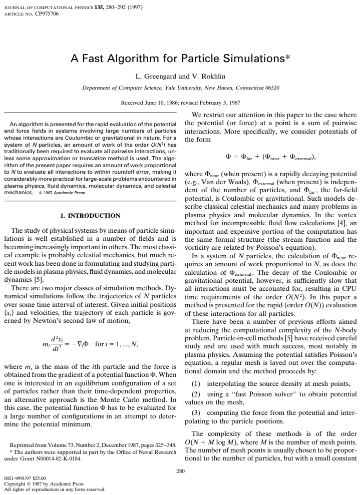
$$\mathbf{u}_\omega(z_j, t) = \sum_{i \neq j} K(z_j - z_i)\Gamma_i,$$

an N-body problem.
To *diffuse*, just add a normally distributed random number,

$$\mathbf{u}_\omega(z_j, t + \Delta t) = \mathbf{u}_\omega(z_j, t) + \sqrt{2\nu\Delta t}\mathcal{N}(0, 1).$$

In practice, there are also redistribution-type methods such that Γ_i is made time-dependent.

Fast multipole method (the *very* short version)



JOURNAL OF COMPUTATIONAL PHYSICS **135,** 280–292 (1997)
ARTICLE NO. CP975706

A Fast Algorithm for Particle Simulations*

L. Greengard and V. Rokhlin

Department of Computer Science, Yale University, New Haven, Connecticut 06520

Received June 10, 1986; revised February 5, 1987

An algorithm is presented for the rapid evaluation of the potential and force fields in systems involving large numbers of particles whose interactions are Coulombic or gravitational in nature. For a system of N particles, an amount of work of the order $O(N^2)$ has traditionally been required to evaluate all pairwise interactions, unless some approximation or truncation method is used. The algorithm of the present paper requires an amount of work proportional to N to evaluate all interactions to within roundoff error, making it considerably more practical for large-scale problems encountered in plasma physics, fluid dynamics, molecular dynamics, and celestial mechanics. © 1987 Academic Press

1. INTRODUCTION

The study of physical systems by means of particle simulations is well established in a number of fields and is becoming increasingly important in others. The most classical example is probably celestial mechanics, but much recent work has been done in formulating and studying particle models in plasma physics, fluid dynamics, and molecular dynamics [5].

There are two major classes of simulation methods. Dynamical simulations follow the trajectories of N particles over some time interval of interest. Given initial positions $\{x_i\}$ and velocities, the trajectory of each particle is governed by Newton's second law of motion,

$$m_i \frac{d^2 x_i}{dt^2} = -\nabla_i \Phi \quad \text{for } i = 1, \ldots, N,$$

where m_i is the mass of the ith particle and the force is obtained from the gradient of a potential function Φ. When one is interested in an equilibrium configuration of a set of particles rather than their time-dependent properties, an alternative approach is the Monte Carlo method. In this case, the potential function Φ has to be evaluated for a large number of configurations in an attempt to determine the potential minimum.

We restrict our attention in this paper to the case where the potential (or force) at a point is a sum of pairwise interactions. More specifically, we consider potentials of the form

$$\Phi = \Phi_{\text{far}} + (\Phi_{\text{near}} + \Phi_{\text{external}}),$$

where Φ_{near} (when present) is a rapidly decaying potential (e.g., Van der Waals), Φ_{external} (when present) is independent of the number of particles, and Φ_{far}, the far-field potential, is Coulombic or gravitational. Such models describe classical celestial mechanics and many problems in plasma physics and molecular dynamics. In the vortex method for incompressible fluid flow calculations [4], an important and expensive portion of the computation has the same formal structure (the stream function and the vorticity are related by Poisson's equation).

In a system of N particles, the calculation of Φ_{near} requires an amount of work proportional to N, as does the calculation of Φ_{external}. The decay of the Coulombic or gravitational potential, however, is sufficiently slow that all interactions must be accounted for, resulting in CPU time requirements of the order $O(N^2)$. In this paper a method is presented for the rapid (order $O(N)$) evaluation of these interactions for all particles.

There have been a number of previous efforts aimed at reducing the computational complexity of the N-body problem. Particle-in-cell methods [5] have received careful study and are used with much success, most notably in plasma physics. Assuming the potential satisfies Poisson's equation, a regular mesh is layed out over the computational domain and the method proceeds by:

(1) interpolating the source density at mesh points,

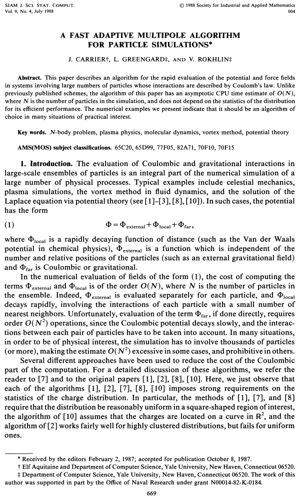
(2) using a "fast Poisson solver" to obtain potential values on the mesh,

(3) computing the force from the potential and interpolating to the particle positions.

The complexity of these methods is of the order $O(N + M \log M)$, where M is the number of mesh points. The number of mesh points is usually chosen to be proportional to the number of particles, but with a small constant

Reprinted from Volume 73, Number 2, December 1987, pages 325–348.

* The authors were supported in part by the Office of Naval Research under Grant N00014-82-K-0184.



SIAM J. SCI. STAT. COMPUT.
Vol. 9, No. 4, July 1988

A FAST ADAPTIVE MULTIPOLE ALGORITHM FOR PARTICLE SIMULATIONS*

J. CARRIER†, L. GREENGARD‡, AND V. ROKHLIN‡

Abstract. This paper describes an algorithm for the rapid evaluation of the potential and force fields in systems involving large numbers of particles whose interactions are described by Coulomb's law. Unlike previously published schemes, the algorithm of this paper has an asymptotic CPU time estimate of $O(N)$, where N is the number of particles in the simulation, and does not depend on the statistics of the distribution for its efficient performance. The numerical examples we present indicate that it should be an algorithm of choice in many situations of practical interest.

Key words. N-body problem, plasma physics, molecular dynamics, vortex method, potential theory

AMS(MOS) subject classifications. 65C20, 65D99, 77F05, 82A71, 70F10, 70F15

1. Introduction. The evaluation of Coulombic and gravitational interactions in large-scale ensembles of particles is an integral part of the numerical simulation of a large number of physical processes. Typical examples include celestial mechanics, plasma simulations, the vortex method in fluid dynamics, and the solution of the Laplace equation via potential theory (see [1]–[3], [8], [10]). In such cases, the potential has the form

$$\Phi = \Phi_{\text{external}} + \Phi_{\text{local}} + \Phi_{\text{far}}, \tag{1}$$

where Φ_{local} is a rapidly decaying function of distance (such as the Van der Waals potential in chemical physics), Φ_{external} is a function which is independent of the number and relative positions of the particles (such as an external gravitational field) and Φ_{far} is Coulombic or gravitational.

In the numerical evaluation of fields of the form (1), the cost of computing the terms Φ_{external} and Φ_{local} is of the order $O(N)$, where N is the number of particles in the ensemble. Indeed, Φ_{external} is evaluated separately for each particle, and Φ_{local} decays rapidly, involving the interactions of each particle with a small number of nearest neighbors. Unfortunately, evaluation of the term Φ_{far}, if done directly, requires order $O(N^2)$ operations, since the Coulombic potential decays slowly, and the interactions between each pair of particles have to be taken into account. In many situations, in order to be of physical interest, the simulation has to involve thousands of particles (or more), making the estimate $O(N^2)$ excessive in some cases, and prohibitive in others.

Several different approaches have been used to reduce the cost of the Coulombic part of the computation. For a detailed discussion of these algorithms, we refer the reader to [7] and to the original papers [1], [2], [8], [10]. Here, we just observe that each of the algorithms [1], [2], [7], [8], [10] imposes strong requirements on the statistics of the charge distribution. In particular, the methods of [1], [7], and [8] require that the distribution be reasonably uniform in a square-shaped region of interest, the algorithm of [10] assumes that the charges are located on a curve in $\mathbb{R}^2$, and the algorithm of [2] works fairly well for highly clustered distributions, but fails for uniform ones.

* Received by the editors February 2, 1987; accepted for publication October 8, 1987.

† Elf Aquitaine and Department of Computer Science, Yale University, New Haven, Connecticut 06520.

‡ Department of Computer Science, Yale University, New Haven, Connecticut 06520. The work of this author was supported in part by the Office of Naval Research under grant N00014-82-K-0184.

Fast multipole method (cont)

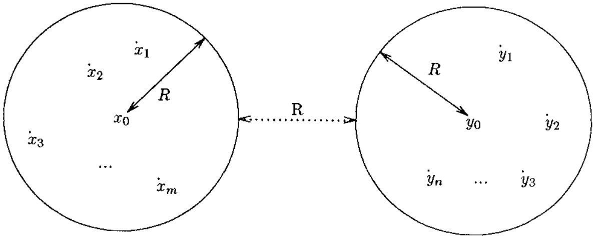


FIG. 1. Well-separated sets in the plane.

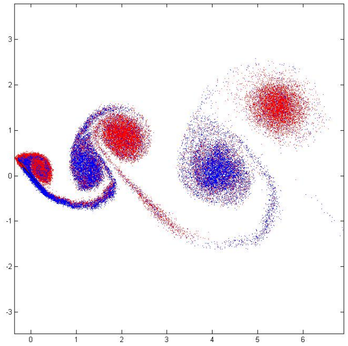
Figure: Found at p. 3 of Greengard and Rokhlin: “A Fast Algorithm for Particle Simulations” *J. Comput. Phys.* **73**(2):325–348 (1987).

Fast multipole method (cont)

Main idea: all charges/potentials/bodies inside two well-separated sets can interact through an operator of low effective rank.
In a nutshell: distribute the points in a recursive tree of boxes where each box has 4 children (2D).

1. *Initialize* at the finest level in the tree, expanding each potential in a *multipole series* around the midpoint of the box.
2. *Go upwards* and *shift* all expansions to parents, yielding a "top expansion" for the whole enclosing box.
3. *Go downwards* and *shift-and-convert* all expansions into *local* expansions (eg. polynomials). Also, *shift* all such expansions to children, yielding a local field in each box.

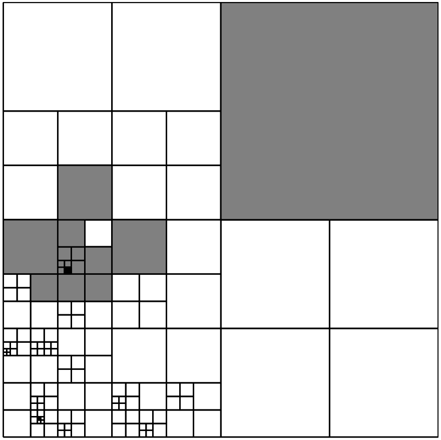
Illustrations



Test: flat plate.
Production run: 3-bladed turbine.

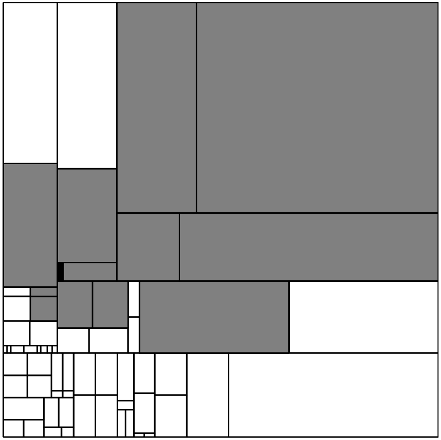
Adaptivity

Want adaptivity, but *quite* complicated... The "C" in $\mathcal{O}(N)$ can be rather large.



Asymmetric adaptivity

Idea: split around the median point instead of around the geometric midpoint. Easier to get the communication localized.



The θ-criterion

As the mesh looses regularity, it becomes important to keep track of what sets are really well-separated.

Criterion

Let the sets $S_1, S_2 \subset \mathbf{R}^D$ be contained inside two disjoint spheres such that $\|S_1 - x_0\| \le r_1$ and $\|S_2 - y_0\| \le r_2$. Given $\theta \in (0,1)$, if $d \equiv \|x_0 - y_0\|$, $R \equiv \max\{r_1, r_2\}$, and $r \equiv \min\{r_1, r_2\}$, then the two sets are *well-separated* whenever $R + \theta r \le \theta d$.

In other words: any of the two sets may be expanded by a factor of $1/\theta$ and arbitrarily rotated about its center point without touching the other set.

Asymmetric adaptivity (cont)

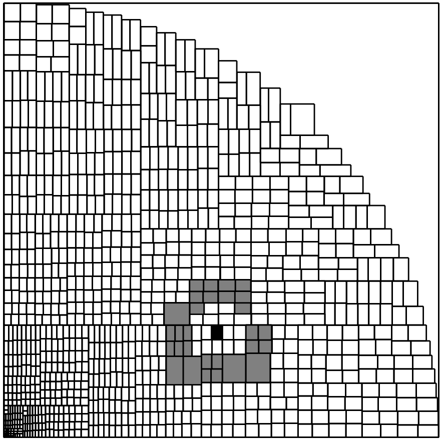


Figure: Typical shift-and-convert interaction list ($\theta = 1/2$).

Accurate?

Theory: the relative error for the pth order adaptive fast multipole method under the θ-criterion is bounded by a constant $\times\ \theta^{p+1}/(1-\theta)^2$.

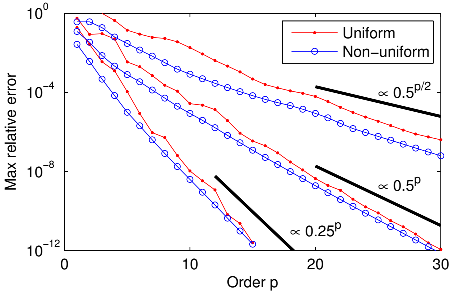


Figure: Errors for two different distribution of points and three distinct θs.

Efficient?

Theory: $\mathcal{O}\left(\theta^{-2}\log^{-2}\theta \cdot N\log^{2}\mathrm{TOL}\right)$. ($\Longrightarrow \theta_{\mathrm{opt}} = \exp(-1) \approx 0.368...$)

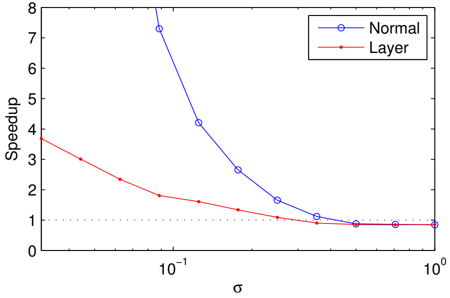
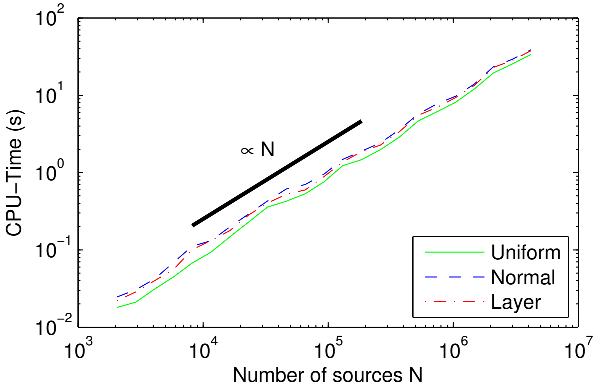


Figure: Adaptive vs. uniform FMM. Two different distribution of points.

"Normal" $:= \mathcal{N}(0, \sigma)$, but rejected to fit within the positive unit square.
"Layer" $:=$ the x-coordinate is $U[0, 1]$ instead.

Robustness: does it scale?



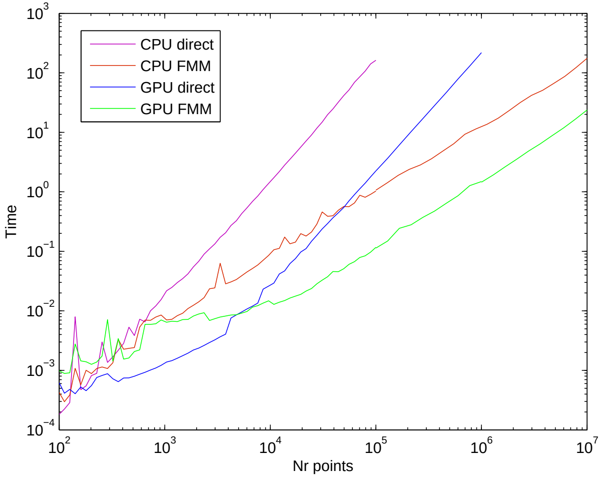
Implementation at the GPU (ongoing...)

- Started with off-loading the perfectly parallel direct evaluation at the finest level.
- Next came the (not so expensive) local shift operation....
- ...and finally the expensive shift-and-convert operation.

What remains serial is currently the initial sorting which is the median-of-three algorithm used in eg. *quicksort*. *It now takes almost 60–80% of the running time!*

GPU Performance

GeForce 480 (**NB** @ 700MHz!)



Conclusions

- Academic sw-project: hardly no planning or overall strategy, just a concrete problem to be solved.
- Nice aspect: the steady and controlled growth of performance and complexity:
  1. Stand-alone *recursive* C99 implementation.
  2. Matlab-interface to a *direct* N-body evaluation.
  3. First working copy; later heavily optimized (eg. BLAS L3).
  4. Added adaptivity – took the time to investigate a novel approach.
  5. Parallel GPU-implementation CUDA/C++ (still ongoing).
  6. 3D...?
- Increasingly sophisticated regression tests.
- In such an academic environment, clarity wrt to goals is *very* important.