高速多重極展開法とツリー法—多体シミュレーションのための高速算法

Fast multipole method and tree method — fast algorithms for many body simulations

牧野淳一郎
東京大学大学院総合文化研究科
東京都目黒区駒場 3-8-1
makino@grape.c.u-tokyo.ac.jp

1998 年 9 月 1 日

英文要旨

I overview the Fast Multipole Method (FMM) and the Barnes-Hut tree method. These algorithms evaluate mutual gravitational interaction between N particles in $O(N)$ or $O(N \log N)$ times, respectively. I present basic algorithms as well as recent developments, such as Anderson's method of using Poisson's formula, the use of FFT, and other optimization techniques. I also summarize the current states of two algorithms. Though FMM with $O(N)$ scaling is theoretically preferred over $O(N \log N)$ tree method, comparisons of existing implementations proved otherwize. This result is not surprizing, since the calcultion cost of FMM scales as $O(Np^2)$ where p is the order of expansion, while that of the tree method scales as $O(N \log Np)$.

1　はじめに

この記事では、多数の質点間の重力相互作用や荷電粒子の間の静電相互作用を高速に計算する手法として1980年代中頃に提案されたFMM (Fast Multipole Method, 高速多重極展開法) [8] と、それと非常に関連が深く、やはり1980年代に提案されて広く使われるようになったツリー法 [2, 3] について、その原理と研究の動向を概説する。まず、基本原理が単純であるツリー法についてその原理を説明し、次にFMMの原理を説明し、その次に、実際に使う上での問題点等についてまとめる。

2　ツリー法の原理

以下、問題を質点系の重力相互作用ということにする。別にクーロン相互作用やそれ以外の（$1/r$ ポテンシャルでないような）相互作用でも、距離が大きくなれば大きさが小さくなるようなものであれば原理が変わるわけではない。また、粒子系以外でも、渦糸法による流体計算、偏微分方程式の境界値問題など、積分形で書けるものには広く応用できる。そういった研究もなされている。（例えば [7]）

解くべき問題は、以下のような常微分方程式系の数値解を求めるということである。

$$\frac{d^2\boldsymbol{x}_i}{dt^2} = G\sum_{j=1,j\neq i}^{N} m_j \frac{\boldsymbol{x}_j - \boldsymbol{x}_i}{|\boldsymbol{x}_j - \boldsymbol{x}_i|^3},$$

ここで N は粒子の数、$\boldsymbol{x}_i, m_i$ はそれぞれ粒子 i の位置、質量であり、G は重力定数である。

右辺を評価するもっとも単純な方法は、右辺をそのまま計算することである。粒子数が数百程度であれば、この方法（単に直接計算という）よりも速くするのは難しい。

しかし、直接計算には $O(N^2)$ で計算量が増えていくという問題がある。計算量を減らすには、遠くの粒子を適当にまとめることが考えられる。例えば遠くの方の粒子のかたまりをそれらの重心で置き換えてもいいし、高い精度が必要なら多重極展開も考えられる。遠くにいくに従ってかたまりを大きくしていけば、直観的には一つの粒子への力を $O(\log N)$ で計算でき、全粒子への力であればその N 倍の $O(N\log N)$ で計算できることになる。

もちろん、ここで問題なのはどうやってうまく粒子をかたまりに分けるかということであろう。ある粒子への力を計算するのに適した分け方を見つけるには、すくなくとも一度は全粒子をみないといけない。従って、その計算量は $O(N)$ より小さくはない。粒子毎に別の分け方をしたら、それだけで計算量が $O(N^2)$ になってしまう。

この問題は、すべての粒子に使える汎用の分け方というのをあらかじめ構成することができれば解決する。それをするのがツリー構造による階層的な空間分割ということになる。

図で説明しよう。3次元の図は書きにくいし見る方も見にくいであろうから、2次元で書くことにする（図1）。まず、平面内の適当な領域に粒子が分布しているとしよう。粒子が有限個なので、そのすべてを含む正方形を考えることが出来る。すべて含んでいればよいので、別に最小である必要はないが、普通はそこそこ小さくとる。これを、まず4つの小正方形 (セル) にわける。で、そのそれぞれをさらに4つにわけるというのを再帰的に繰り返していく。

粒子が0個または1個しかなくなったところで止める。この空間（平面）分割に対応したツリー構造を考えると、根に全体の正方形に対応する節点があり、その下に4つの小正方形、さら

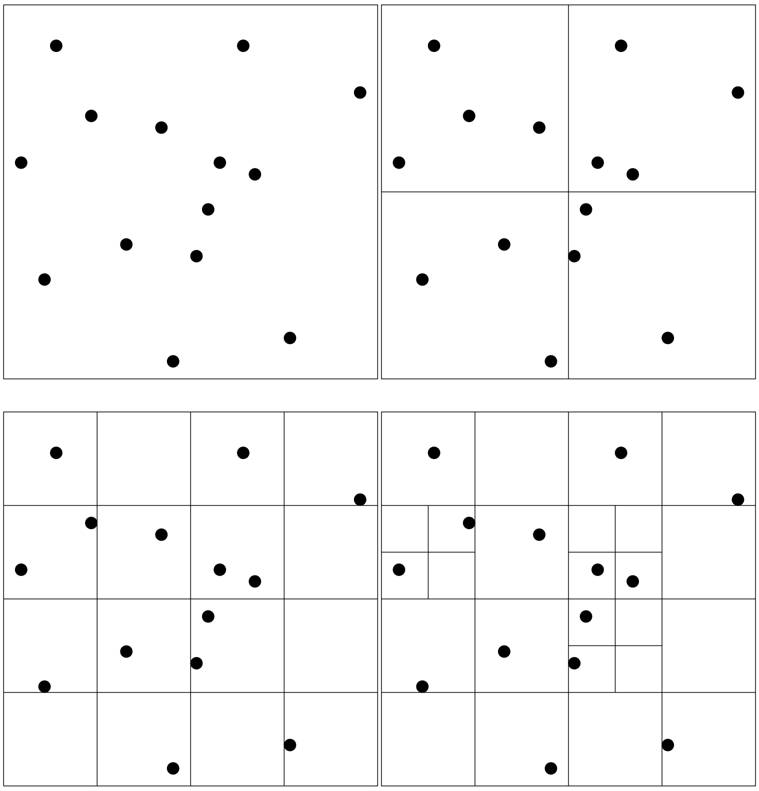

図 1: 4 分木の構築

にその下にそのそれぞれの中の 4 つ$\cdots$ と続いていって、ツリーの葉には粒子 1 つ 1 つがくることになる。この方法では、粒子の数密度が高いところでは細かく、そうでないところでは粗いという意味で、適応的な構造が実現されていることに注意してほしい。3 次元にすれば、正方形 4 個の代わりに立方体 8 個となるが、それ以外は全く同じである。

まず、どうやってこのツリー構造を使って一つの粒子への重力が計算できるかということを考えてみる。このためには、ツリーのある節点が表す立方体内の粒子全体からのある粒子への力を計算する方法を与えればよい。(なお、以下では、節点に対応する立方体、あるいはそのなかの粒子全体のことも単に節点ということがある) 系全体からの力は、単に根節点からの力として計算できる。一つの節点からの力を以下のように計算することにする

$$\text{ある節点からある粒子への力} = \begin{cases} \text{その節点の重心からの力} \\ \text{(節点と粒子が「十分離れている」)} \\ \text{その節点の子節点からの力の合計} \\ \text{(それ以外)} \end{cases}$$

「十分離れている」かどうかの判定には通常は以下のような基準を使う。

$$\frac{l}{d} < \theta$$

ここで d は粒子と節点の重心の距離、l は節点に対応する立方体の一辺の長さである。パラメータθは見込み角といわれる量であり、計算精度と計算量を制御する。高い精度を得たければ θ を

小さくすればいいが、漸近的には θ^{-3} に比例して計算量が増える。計算精度をあげるもう一つの方法は、重心だけを考えるのでなく多重極展開を作っておくことである。

原点中心で半径 a の球内にある粒子が球の外に作るポテンシャルの多重極展開係数は

$$\alpha_l^m = \sum_{i=1}^{N} m_i \left(\frac{r_i}{a}\right)^l Y_l^{-m}(\theta_i, \phi_i)$$

で与えられる。ここで、m_i と (r_i, θ_i, ϕ_i) はそれぞれは粒子 i の質量と極座標で表した位置である。また、$Y_l^m(\theta, \phi)$ は l 次の球面調和関数で、ルジャンドル陪関数 P_l^m を使って

$$Y_l^m = (-1)^m \sqrt{\frac{2l+1}{4\pi}\frac{(l-|m|)!}{(l+|m|)!}} P_l^m(\cos\theta) e^{im\phi}$$

と書ける。この展開係数を使うと、位置 (r, θ, ϕ) （$r > a$）でのポテンシャルは

$$\Phi(r, \theta, \phi) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \alpha_l^m \frac{a^l}{r^{l+1}} Y_l^m(\theta, \phi).$$

と書けることになる。

ツリー法を実際に使うには、各節点についてその中の粒子全体の重心の位置と質量（あるいは多重極展開の展開係数）をあらかじめ求めておく必要がある。これらはやはり再帰的に定義することができる。つまり、子節点の中の粒子全体の重心の位置と質量がわかっていれば、親節点の情報は計算できる。多重極展開の展開係数についても（計算は繁雑であるが原理的には）おなじことである。実際の計算プログラムにおいては、再帰手続きで実現してもよいが並列化などを考えると葉節点から根節点に向かって順に計算していくほうがよい。

3　FMMの原理

ツリー法の原理から FMM の原理にはほんの一歩である。ツリー法では、ある粒子への力を計算するのに、遠くの粒子からの力はまとめて計算することにした。相互作用は対称的であるので、逆にある粒子から遠くの粒子への力をまとめて計算するということも考えられる。両方を同時にやれば、ある一群の粒子から別の一群の粒子への力をまとめて計算することになる (図 2)。このように、受ける側もまとめてやることによって、以下に説明するようにツリー法の $O(N \log N)$ から $O(N)$ へ計算量のオーダーを下げることができる。このためにこの方法には「高速」多重極展開法という名前がついている。

力を受けるほうは、重力ポテンシャルのテイラー展開を求めておいて、それを各粒子の位置で評価するという操作が必要になる。もちろん、実際にはポテンシャルは調和関数になっているので、球面調和関数で展開することで項数をテイラー展開に比べて減らすことができる。この展開のことを、以下局所展開と呼ぶ。

この方法は、Rokhlin と Greengard によってまず 2 次元の場合に提案され [8]、すぐに 3 次元に拡張された [9]。考え方は 2 次元でも 3 次元でも同じであるので、例によって図は 2 次元で示す（図 3）。ツリー法では適応的なツリー構造を扱ったが、FMM ではとりあえず一様なツリー構造を考える。すなわち、ツリーのレベルを k とすると、元の正方形を 4^k 個の小正方形に分割

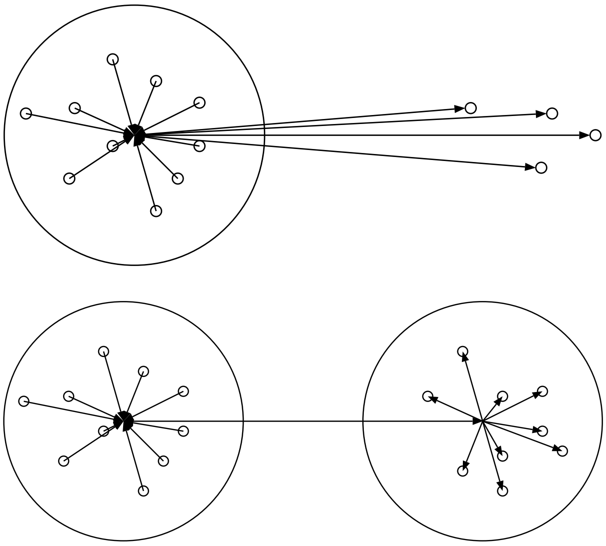

図 2: ツリー法（上）とFMM（下）

する。なお、適応的なツリー構造の場合のアルゴリズムは例えば [4] をみられたい。ツリー法の場合と同様に、あらかじめ各セルに含まれる粒子（一つも無いこともありえるが）が作るポテンシャルの多重極展開は準備しておく。

FMM では、一つの粒子への力を以下のように分割する

$$\boldsymbol{F} = \boldsymbol{F}_d + \boldsymbol{F}_c$$

ここで、$\boldsymbol{F}_d$ は直接計算する分で、自分のいるセルと、それに隣接したセルの粒子からの力である。これらには多重極展開は使えないので、直接計算することになる。残るのが $\boldsymbol{F}_c$ であるが、これはさらにツリーのレベル毎に別に計算する。つまり、あるセル i に対し、(i) そのセルと同じレベルにあって、(ii) その親セルとその隣接 8 セルの合計 9 セルの子であって、(iii) 問題のセルには隣接していない、$27\ (= 36 - 9)$ 個のセルを考える。この 27 個のセルに含まれる粒子が作る重力ポテンシャルの局所展開の和を、セル i の中心で評価しておく。トップレベル（系全体）と、そのすぐ下のレベルでは、親のレベルでは隣接していて自分のレベルでは隣接していないものは存在しないので、計算の必要はない。

このようにして各レベルで重力場を計算しておくと、あとは粒子の位置で、それぞれのレベルでの局所展開を評価して合計すればいい。が、別に粒子毎に全レベルでの展開を評価しなくても、上のレベルの展開は一段下に移してやって（展開の中心をシフトして）まとめておけば、粒子は全部まとめた局所展開を一度だけ評価すればいい。この、局所展開をシフトしながら下に落していくのは、最初に多重極展開をやはりシフトしながら上にあげていくのとちょうど対称的な操作になる。

図 4 のように、4 レベルのセルの中心での展開を合計する場合を考えてみる。まず、一番上のレベルのセルの中心での局所展開を、4 つの子セルの中心に展開中心をシフトしてそれぞれのレベルでの重力場の局所展開と足し合わせる。さらに、それぞれの子セルで、合計した局所展開をまたそれらの 4 つの子セルの中心にシフトし、またそこでの局所展開と足し合わせる。これを

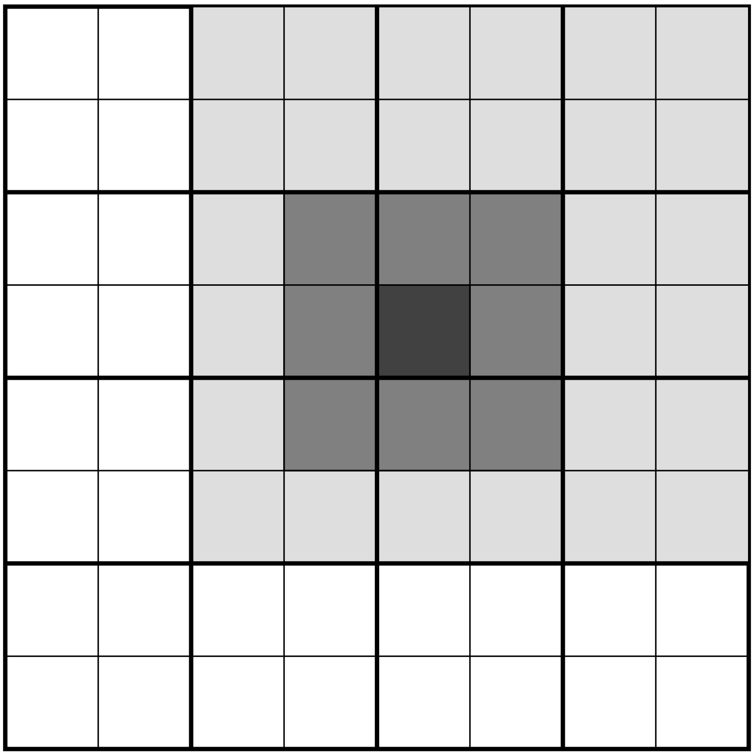

図 3: FMM の概念。もっとも濃く塗ったセルに入っている粒子への力を評価する。自分とその回りの 8 個のセルからの寄与は直接計算し、さらにその回りの 27 個は多重極展開を使う。その上のレベルは親セルが面倒を見る。

最下層のセルに達するまで繰り返すと、各セルの中心での、自分と自分に隣接するセルに入っている粒子を除いた他のすべての粒子による力の局所展開がもとまる。粒子一つへの力は、この最下層での局所展開を粒子の位置で評価したものと、隣接セルの粒子からの力の合計ということになる。

ここで計算量のオーダーを考えてみると、すべての操作の計算量が粒子数またはセルの数にしか比例していないことがわかる。セル数は粒子数に比例するようにとれるので、結局計算量のオーダーが $O(N)$ ということになるわけである。

4　他の方法との関係

さて、粒子間相互作用を高速に計算する方法は、ツリー法や FMM の他にもいろいろある。ここでは、それらとの関係や、適切な応用について簡単に触れておこう。

多数の粒子が作るポテンシャル場を高速に計算する方法としてもっとも広く使われているのは、空間格子を切って FFT を使う PM 法 [10] (particle-mesh 法) であろう。これはほかの方法に比べて圧倒的に高速であり、計算精度や空間分解能がそれほど高い必要がないときには最適である。

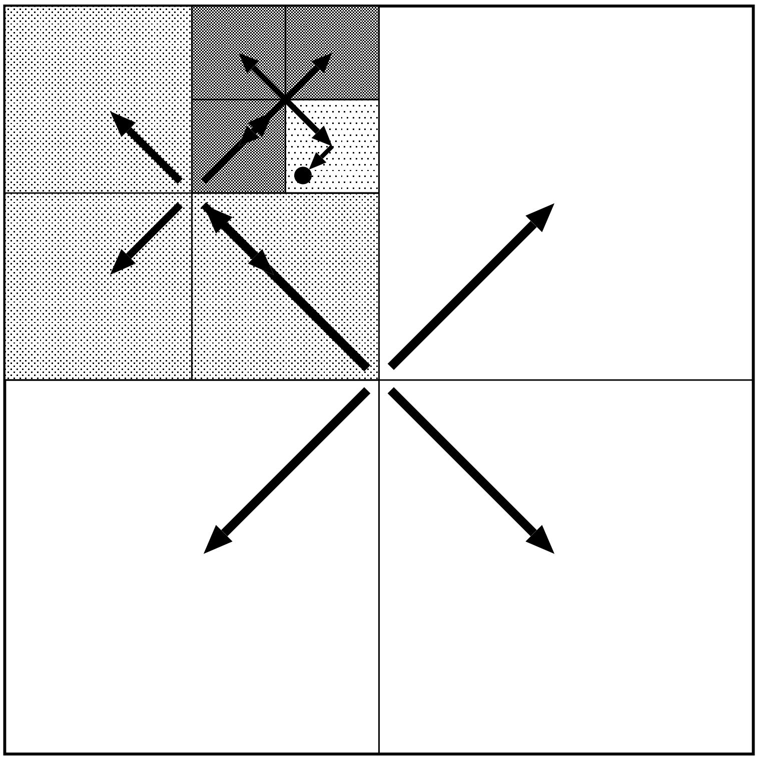

図 4: 局所展開のシフトと加算

フーリエ変換を使いつつ精度や分解能を上げる方法としては、Ewald 法や P^3M[10] がある。どちらも近くは直接計算、遠くはフーリエ空間で計算する。違いはフーリエ変換に FFT を使うかどうかである。Ewald では FFT を使わないので、計算量は増えるが離散化誤差がなくなる。これらの方法の問題点は、粒子分布が一様からずれると急速に計算量が増えることと、周期境界条件を置かない場合には計算が面倒になることである。

これに対し、ツリー法や FMM の場合には、適応的なツリー構造を扱うプログラムを実現することは比較的容易であり、粒子分布の性質が悪くてもそれほど計算量が増えない。計算法自体に、ツリーが一様かどうかに依存するところがないためである。これはこれらの方法の大きな特徴である。なお、FFT を使う方法とは逆に、周期境界の場合の扱いは面倒になる [13]。

5 FMM の実装と高速化

5.1 アンダーソンの方法

FMM を実装する上での大きな問題の一つは、定式化、特に球面調和関数のシフトや多重極展開から局所展開への変換が複雑であり、プログラムの開発や高速化が難しいということであった。Anderson [1] はこの困難を解決する画期的な方法を提案した。この方法を使うとプログラムが簡単になるだけでなく実行速度の観点からもメリットがあるという結果が得られている [5]。

以下、この方法を簡単に紹介しよう。

基本的なアイディアは、球面調和関数の展開係数をデータとして使う代わりに、球面でのポテンシャルの値そのものを使おうというものである。半径 a の球の中の粒子が外に及ぼすポテンシャルは、球の表面でのポテンシャルの値がわかっていればラプラス方程式の境界値問題を解けば与えられる。解はポアソンの公式を使って以下のように書ける

$$\Psi(\boldsymbol{x}) = \frac{1}{4\pi}\int_S \left[\sum_{n=0}^{\infty}(2n+1)\left(\frac{a}{r}\right)^{n+1}\mathrm{P}_n(\boldsymbol{s}\cdot\boldsymbol{x}/|x|)\right]\Psi(a\boldsymbol{s})ds,$$

ここで P_nは n 次ルジャンドル関数である。この積分を、球面上での適当な標本点を使った数値積分に置き換える。2 次元であれば、円を等分割すればいいが、3 次元では点のとり方に工夫がいる。また、エリアシングによる誤差を防ぐために点数に応じて中の無限和を適当なところで打ち切る必要がある。

子セルの外の多重極展開をその一つ上のセルでのそれに変換するには、上の式に従って親セルを含む球面上でのポテンシャルを求めればいい。また、局所展開も、$1/r$ での展開が r での展開に変わるだけで上と同じ式である。さらに、多重極展開を局所展開に変換するには、上の式をつかって局所展開する球面上でポテンシャルを評価するればよい。つまり、上の式だけで、FMM に必要な数学がすべて表現されていることになる。

5.2 直接計算とのトレードオフ

多重極展開を局所展開に変換する部分の計算コストは非常に大きい。展開次数を p とすると 3 次元ではこれが Np^4 に比例し、 p を大きくすると計算量のほとんど全部になる (多重極展開や局所展開のシフトも計算量のオーダーは同じだが、係数が 2 桁ほど小さい)。このため、他のところの計算量を増やしてでもここを減らせれば大きな利得になる。

セル数を減らせばこの計算量は減る。そのかわり、最下層のセルに入っている粒子数が増えるので、直接計算のところの計算量が増えることになる。この 2 つがバランスするようにセル数を決めると、セル当たりの粒子数を p^2 に比例するようにとればいいことがわかる。このとき、実効的な計算量のオーダーは $O(Np^2)$ になる [4]。

5.3 FFT による高速化

多重極展開から局所展開への変換やシフトは、展開係数のベクトルに変換行列を掛ける操作になる。 Greengard と Rokhlin [9] はこの変換を FFT を使って高速化できることを示唆した。Elliott と Board はこの方法を実現し、詳細な性能評価を行なった [6]。例えば $p = 16$ までとったときには変換自体は 3 倍ほど高速化されている。ただし、上に述べた直接計算とのトレードオフがあるので、計算全体の高速化は 1.7 倍程度ということになる。

5.4 並列化

並列化はツリー法、FMM の両方で良く研究されている。どちらでも、どのように空間分割をして粒子と領域をプロセッサに割り付けるかが問題となる。逆にいえば、それが決まれば並列化

表 1: 実行速度の例。数値は計算時間（秒）

粒子数	16K	64K	256K	1024K
FMM	1.9	7.5	28.5	103.5
Anderson	1.3	5.0	17.9	67.8
BH tree	0.57	2.7	12.4	54.0

そのものはそれほど複雑ではない。メッセージパッシングの機械では Warren と Salmon による ORB（再帰的直交分割）や Morton ordering による領域分割が良く知られている [17, 18]。物理的に共有メモリである機械や、分散共有メモリの場合は、Morton ordering によって粒子をソートしておくことで自動的に空間分割が実現でき、かなり良い性能が得られている [11]。また、最近は HPF などのデータ並列型の言語を使った並列化も試みられている [12]。

6　計算速度の例

ここでは、比較的最近の Blackston and Suel[5] の結果を紹介しよう。彼らは多重極展開を使った FMM, アンダーソンの方法を使った FMM、および普通のツリー法を、様々な並列計算機上に実装し、その精度、性能を比較した。結果のすべてを紹介は出来ないので、比較的精度が低い（ポテンシャルの相対精度が 10^{-3}程度）の場合に粒子数と 1 ステップの計算時間の関係を表 1 に示す。粒子分布はプラマーモデルと呼ばれる球対称なもので、計算機は 16 プロセッサの Cray T3E である。ここではアンダーソンの方法の時間が示されていないが、別のデータでは FMM に比べて 2-3 倍高速となっている。すなわち、FMM はツリー法に比べて有意に遅いが、アンダーソン法ではほぼ同等な計算速度を実現できる。

FMM は Anderson 法の場合でもツリー法に比べてはるかに複雑である。それだけの努力を必要とするにもかかわらず、現在のところ計算精度が同程度の場合に FMM のほうが速いという報告はないということは心に留めておくべきであろう。

なお、上の議論は 3 次元の場合である。FMM の計算量は 2 次元と 3 次元で大きく違うということに注意しておく。2 次元では調和関数展開の項数は $O(p)$ であり、また相互作用するセルの数が通常の FMM の場合で 27 と小さい。これに対して 3 次元では項数が $O(p^2)$ となり、また相互作用するセル数が 189 と大きくなるためである。

7　いくつかの話題

7.1　多体問題専用計算機との関係

筆者らのグループでは、粒子間相互作用を計算することだけに特化した計算機を開発し、実際に天文学の研究に使っている [15, 16]。FMM やツリー法のような速い方法があるのに、こういった計算機を作ることに意味があるのだろうか。

我々が専用計算機というアプローチをとる理由は 2 つである。一つは、粒子毎にタイムスケールが大きく違うような問題では、ツリー法や FMM を使うのは非常に難しい、特に並列計算機

で使えるような実現法はまだ存在しないということである。

もうひとつの理由は、ツリー法や FMM も、実は直接計算の部分を高速化するだけでかなり速くなるということである。5.2 節で述べたように、多重極展開の計算量と直接計算の計算量の間にはトレードオフが働く。このために、粒子間相互作用の計算が例えば 100 倍高速になれば、セルのなかの粒子数を 10 倍にすることで計算全体を 10 倍速くすることができる。つまり、粒子間相互作用の計算速度の平方根に比例して計算速度が改善される。ツリー法でも事情はほぼおなじであり、実際に 10-30 倍程度の加速が実現されている [14]。

7.2 計算精度に対する要求について

実用的にも理論的にも大きな問題である、「計算精度をどう評価するか」ということについて簡単に触れておきたい。

FMM についての論文では、適当な粒子分布を与えてポテンシャルや加速度を計算し、その最大値とか平均値を使って評価する。しかし、時間発展を追う問題として考える時には、最終的に小さくしたいのは積分して求まった各粒子の軌道の誤差である。

もちろん、軌道は加速度を積分してもとまるのだから、加速度の誤差が小さければ軌道の誤差は小さい。しかし、軌道の誤差には加速度の誤差の時間相関が効いてくる。つまり、仮に誤差が大きくても、タイムステップ毎にランダムに変わるのであれば、小さくてもいつも同じ方向に働くのに比べて積分した誤差は小さくなるかもしれないわけである。

例えば、系の中を粒子がランダムに動き回っているような場合であれば、近くの粒子からの力の誤差の相関時間は短く、遠くからのは長いであろう。逆に、粒子がほとんど止まっていれば、すべての誤差の時間相関が長い。実際の精度評価は、このような対象系の特性を考えに入れて行なう必要がある。ある瞬間での力の誤差を最小にするような方法がもっともよいとは限らないのである。

参考文献

[1] Anderson C. R., An implementation of the fast multipole method without multipoles. *SIAM Journal on Scientific and Statistical Computing* 13 (1992), 923–947.

[2] Appel A. W., An efficient program for many-body simulation. *SIAM Journal on Scientific and Statistical Computing* 6 (1985) 85–103.

[3] Barnes J. and Hut P., A hiearchical $O(N \log N)$ force calculation algorithm. *Nature* 324 (1986), 446–449.

[4] Board J. A. J., Hakura Z. S., Elliott W. D., and Rankin W. T., Scalable variants of multipole-based algorithms for molecular dynamics applications, *Proceedings of the Seventh SIAM Conference on Parallel Processing for Scientific Computing* (edited by Bailey *et al.*), SIAM, Philadelphia, 1995, 295–300.

[5] Blackston D. and Suel T., Highly portable and efficient implementations of parallel adaptive n-body methods. *Proceedings of SC97.* ACM 1997, (CD–ROM).

[6] Elliott W. D. and Jr. J. A. B., Fast fourier transform accelerated fast multipole algorithm. *SIAM Journal on Scientific Computing* 17 (1996), 398–415.

[7] 福井卓雄、服部純一、土居野優, 高速多重極法の境界要素解析への応用. 構造工学論文集 43A (1997), 373–382.

[8] Greengard L. and Rokhlin V., A fast algorithm for particle simulations. *Journal of Computational Physics* 73 (1987), 325–348.

[9] Greengard L. and Rokhlin V., Rapid evaluation of potential fields in three dimensions. *Vortex Methods* (edited by Anderson C. and Greengard, C.), number 1360 in Lecture Notes in Mathematics, Springer-Verlag, Berlin, 1988, 121–141.

[10] Hockney R. W. and Eastwood J. W., *Computer Simulation Using Particles.* IOP Publishing, Ltd., Bristol, 1988.

[11] Holt C. and Singh J. P., Hierarchical N-body Methods on Shared Address Space Multiprocessors, *Proceedings of the Seventh SIAM Conference on Parallel Processing for Scientific Computing* (edited by Bailey *et al.*), SIAM, Philadelphia, 1995, 313–318.

[12] Hu Y., Jonsson S. L., and Teng S.-H., A data-parallel adaptive n-body method, *Proceedings of the Eighth SIAM Conference on Parallel Processing for Scientific Computing.* SIAM, 1997, (CD-ROM).

[13] Lambert C. G., Darden T. A., and Board J. A., Jr., A multipole-based algorithm for efficient calculation of forces and potentials in macroscopic periodic assemblies of particles. *Journal of Computational Physics* 126 (1996), 274–285.

[14] Makino J., Treecode with a special-purpose processor. *Publications of the Astronomical Society of Japan* 43 (1991), 621–638.

[15] Makino J. and Taiji M., *Special Purpose Computers for Scientific Simulations – The GRAPE systems.* John Wiley and Sons, Chichester, 1998.

[16] Sugimoto D., Chikada Y., Makino J., Ito T., Ebisuzaki, T. and Umemura M., A special-purpose computer for gravitational many-body problems. *Nature* 345 (1990), 33-45.

[17] Warren M. S. and Salmon J. K., Astrophysical N-body simulations using hierarchical tree data structures, *Supercomputing '92*, IEEE Comp. Soc., Los Alamitos, 1992, 570–576.

[18] Warren M. S. and Salmon J. K., A parallel hashed oct-tree N-body algorithm, *Supercomputing '93.* IEEE Comp. Soc., Los Alamitos, 1993, 12–21.