Chapter 1
Comparison of an $O(N)$ and an $O(N \log N)$ N-body solver

Gavin J. Pringle*

Abstract

In this paper we compare the performance characteristics of two 3-dimensional hierarchical N-body solvers: an $O(N)$ and an $O(N \log N)$ solver. We present the execution times for numerous N-body force evaluations using the two methods, with various values of N and ϵ, where ϵ is the prescribed error. We find that the $O(N \log N)$ method is more suited to problems which demand a high precision and large N. We then consider how parallelisation affects the algorithms' relative performance.

1 Introduction

The N-body problem consists of a collection of N particles each exerting a force upon one another. The Nth particle is acted upon by the remaining $(N-1)$ particles, hence the time to compute the force acting on each particle is $O(N^2)$. There are a collection of N-body solvers which reduce the time to compute the N-body problem by introducing approximations. In this paper we compare the performance characteristics of two 3-dimensional hierarchical N-body solvers; an $O(N)$ and an $O(N \log N)$ solver.

Our $O(N)$ method derives from the Greengard-Rokhlin Fast Multipole Method (FMM) [5, 6]. Examples of other $O(N)$ N-body solvers may be found in [17, 1, 10]. The $O(N \log N)$ method utilises the same framework as the FMM, but in a manner which is analogous to the Barnes-Hut Algorithm (BHA) [3]. Both the FMM and the BHA are readily parallelised and have been implemented on a wide range of parallel computers [4, 7, 8, 9, 14, 15, 16, 17].

The basic notion behind these algorithms is that a cluster of particles is replaced by a single source, described by a multipole expansion. The force field exerted by the cluster can be approximated by the force field exerted by this multipole source, provided that the distance between the point of evaluation and the cluster is large enough. Moreover, as the distance to the cluster increases, we may increase the radial size of the multipole source. This idea is effected by delineating the clusters with a hierarchy, or 'tree', of cells.

The difference in the order of the two algorithms lies in the manner in which the hierarchy of cells is utilised. In the $O(N \log N)$ method, the interaction model may be described as a 'particle-cell interaction' model, that is, the force exerted on each particle is approximated by its interaction with the multipole sources contained in the cell hierarchy. For the $O(N)$ method, however, the interaction model may be described as a 'cell-cell interaction' model. This is not strictly true, as the sources contained in the cell hierarchy do not actually interact, but this nomenclature does elucidate the character of the method. In this case, a local expansion is created for every cell in the hierarchy, in terms of the multipole expansions.

*Mathematics Department, Napier University, Edinburgh, EH14 1DJ, SCOTLAND

It would appear that this $O(N)$ method is quicker than the $O(N \log N)$ method; however, time for execution also depends strongly on the precision required by the calculation and on other specific implementation details.

In this paper we present the execution times for numerous N-body force evaluations using the two methods, with various values of N and ϵ, where ϵ is the prescribed error. Care is taken to ensure that all other parameters are optimised with respect to N and ϵ. These include tree depth and the number of terms taken in each multipole expansion. From the resulting execution times, we are able to determine which of the two methods, the $O(N)$ or $O(N \log N)$ method, is faster for a given N and ϵ. We then consider how parallelisation affects the algorithms' relative performance, as the larger memory resource allows for an extended N-ϵ space.

2 Informal description of the two solvers.

This section attempts to give a rough illustration of both of the N-body solvers. A fully detailed description of the $O(N)$ solver, and the relevant mathematical operators, may be found in [10].

Both the $O(N)$ method and the $O(N \log N)$ method utilise the same hierarchy of multipole expansions, which is described in the following section.

2.1 The hierarchy of multipole expansions

Particles may be grouped together into clusters and represented by a list of coefficients which describes their distribution, namely a **multipole expansion**. Consider the case of interacting gravitational particles, as shown in figure 1. Suppose we have a cluster of

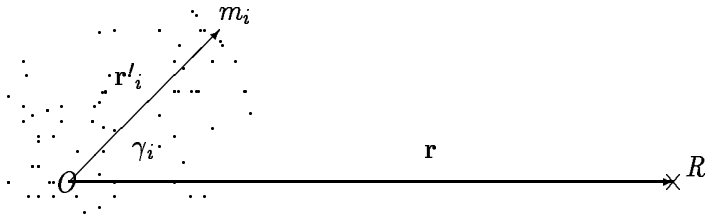

FIG. 1. *A cluster of particles.*

particles of mass m_i at positions $\mathbf{r}'_i$ from the origin. If $|\mathbf{r}| > |\mathbf{r}'_i| \ \forall i$, then the scalar potential, Φ, at a point of evaluation, R, located at position $\mathbf{r}$ is given by

$$\Phi(R) = -G \sum_i \frac{m_i}{|\mathbf{r} - \mathbf{r}'|} = \frac{-G}{|\mathbf{r}|} \sum_i \sum_{n=0}^{\infty} m_i \left(\frac{\mathbf{r}'_i}{\mathbf{r}}\right)^n P_n(\cos\gamma_i), \tag{1}$$

where $P_n(x)$ is the Legendre polynomial of degree n, γ_i is the angle subtended between the vectors $\mathbf{r}$ and $\mathbf{r}'_i$, and G is the gravitational constant. We assume that G=1 without loss of generality. Eqn.(1) describes the potential as a multipole expansion centred about the origin.

If we ensure that all the particles lie within a sphere of radius a, i.e. $|\mathbf{r}'_i| < a$, and insist that $|\mathbf{r}| = ca$, for some $c > 1$, then $\left|\frac{\mathbf{r}'_i}{\mathbf{r}}\right| < \frac{1}{c}$. If p is the highest order retained in the multipole expansion, then it may be shown that the truncation error is given by

$$\epsilon_{abs} = \left| \frac{1}{\mathbf{r}} \sum_i \sum_{n=p+1}^{\infty} m_i \left(\frac{1}{c}\right)^n P_n(\cos\gamma_i) \right| \leq \frac{A}{|\mathbf{r}|(c-1)} \left(\frac{1}{c}\right)^p, \tag{2}$$

where $A = \sum_i |m_i|$. Hence the value of p required to achieve a given relative precision, $\epsilon = \frac{|\mathbf{r}|\epsilon_{abs}}{A}$, may be calculated from

$$p = \lceil -\log_c(\epsilon(c-1)) \rceil, \tag{3}$$

where c is calculated in terms of the distance to a point within the cluster and the cluster's radius, i.e. $c = \frac{|\mathbf{r}|}{a}$. This is described in full in [13].

As the distance to a cluster of bodies increases, then the radius of this multipole expansion may also increase. This idea is effected by delineating the clusters with a hierarchy of cells. The cells are constructed by recursively subdividing the computational domain. In 3 dimensions, the entire domain is enclosed by a cube which is then subdivided into eight equal cubes. This subdivision is performed recursively until there is only a small number of particles per cell. The top level is labelled level 0; hence at any particular level l there are 8^l cubes; this is known as a **oct-tree**.

The total number of levels, n, employed by the mesh has a strong influence over the execution time and is dependent on N, p and the distribution of particles [10, 12].

At this point, it is necessary to introduce some terminology which is relevant to both algorithms. At any level of refinement l, a cell x is subdivided into 8 cells, which are located at level $l+1$. The 8 cells are known as the **children** of x; x is known as the **parent**. Cells which have no children are known as **leaf cells**. Cells which lie at the same level of refinement as cell x are known as **near-field** cells, provided that the centre of these cells lies less than a radial distance of $3d$ away from the centre of cell x, where d is the length of one side of a cube. This gives a maximum of 92 near-field cells. The **interaction set** of a cell x is defined as those cells outwith the near-field cells, which are the children of the neighbours of x's parent.

For both our N-body solvers, a truncated multipole expansion is created for every cell in the hierarchy that contains at least one particle. The method of determining the value of p required to achieve a given precision is similar to that employed in the FMM. In this case the multipole expansions are centred on the geometric centre of each cell in the hierarchy. The BHA, on the other hand, centres the multipole expansion at the cluster's 'centre-of-mass'. This latter system is beneficial for problems where the 'strengths' of the particles are all positive. When this is the case, the dipole moment is identically zero, thus if only the first term is taken from the multipole expansion, the error behaves as if both the first and the second terms were taken. However, many N-body simulations, such as Vortex Methods in CFD, require both positive and negative 'strengths'. Moreover, the error which arises from the use of the latter system is typically much smaller than the prescribed error in practice. This is due to the fact that the least upper bound to this error must be calculated in terms of the worst possible distribution of particles, unlike the FMM [13]. Thus one may predetermine the run-time error with greater control if the multipole expansion is centred on the centre of the cell.

The hierarchy of cells is created by first forming multipole moments for each leaf cell in terms of the particles which lie therein. The Legendre polynomial in eqn.(1) is expanded exactly in terms of spherical harmonics via the Addition Theorem [6]. The multipole moments are given in terms of these spherical harmonics. Multipole moments are not calculated for empty cells. The tree is then traversed to the top, level=0, creating multipole moments for each cell in terms of the multipole moments associated with the 8 child cells. This is performed systematically and efficiently, since the multipole expansions are centred on the cells' centres.

2.2 Informal description of the $O(N \log N)$ solver.

The $O(N \log N)$ solver proceeds as follows. Each cell in the tree is considered in turn, starting at the coarsest level, level=0. If the cell is not in the near-field of the cell which contains the particle, then the associated multipole moments are used to approximate the potential. The next level of refinement is then considered, where all cells which are not part of the near-field, and which have not yet been accounted for, will contribute their associated expansions. For any cell x, the set of cells which contribute a potential to particles in cell x, is the interaction set of cell x. Thus the interaction set need only be located once per cell, and not once per particle. Once at the finest level, the only particles which have not yet contributed to the potential will be the particles which reside in the near-field leaf cells. The pairwise interactions between these particles are summed directly.

Consider a cell in the hierarchy, cell i say, which is to contribute an approximated potential to a particular particle. We calculate the distance, r_i say, between the particle and the centre of that cell, and the radius of the sphere which circumscribes it, a_i say. By eqn.(3), and since the multipole expansion is centred at the geometric centre of the cell, we require p_i terms to achieve a certain relative precision, ϵ, such that

$$p_i = \lceil -\log_{c_i}(\epsilon_i(c_i - 1)) \rceil,$$

where $c_i = \frac{r_i}{a_i}$. Thus, the more distant a cell in an interaction set, the fewer terms will be required to achieve a specific precision.

2.3 Informal Description of the $O(N)$ solver

The $O(N)$ solver is identical to the $O(N \log N)$ solver up to the point where the multipole moments are calculated for every cell in the hierarchy. At this stage, in the case of the $O(N)$ solver, local expansions are created for every cell in the tree (starting at the root cell, level=0). A cell's associated local expansions describes the potential in that cell due to the particles which lie outwith itself and its near-field cells. The local expansion of cell x is formed from the multipole moments associated with all the interaction set of cell x, and from the local expansion moments associated with the parent cell of cell x. Local expansions are not computed for empty cells. Once at the finest level, each leaf cell has an associated local expansion, which is then evaluated at each of the particles which lie therein. As with the $O(N \log N)$ solver, the 'direct' pairwise summation method is used to evaluate the potential due to particles which lie in the near-field leaf cells.

When employing the multipole moments to form the local expansions, only p_i multipole moments are required, where

$$p_i = \lceil -\log_{c_i}(\epsilon(c_i - 1)) \rceil, \tag{4}$$

where $c_i = \frac{\rho_i}{a} - 1$, where ρ_i is the distance between the centre of the cell associated with the local expansion, and the centre of the cell associated with the multipole expansion [10, 13]. Each local expansion has p terms, where $p = \max_i(p_i)$. Note that, for both methods, the same value of p is required to achieve the prescribed precision.

The symmetry inherent in the oct-tree is exploited to reduce the amount of computation. When a local expansion is used to form the local expansions of a cells 8 child cells, only one local expansion is formed. The remaining 7 local expansions are formed by multiplying the first local expansion by a shifting vector. This technique reduces the operation count for this operation from $O(8p^4)$ to $O(p^4 + 8p^2)$. Another element of symmetry is exploited. If cell j lies in the interaction set of cell i say, then the opposite is also true; in set notation,

cell $i \in$ int(cell j) $\Rightarrow$ cell $j \in$ int(cell i). Thus, once the interaction set cells have been located, their associated local expansions may also be calculated at the same time. This is similar to the pairwise interaction used in the 'direct' method and reduces this computation by a factor of order 2.

The inherent symmetry of spherical harmonics is also utilised. If cell $i \in$ int(cell j) and cell i lies directly above, or directly below cell j, then the computation is reduced substantially. Moreover, for both methods presented in this paper, this symmetry is also used to reduce (i) the amount of memory one requires, (ii) the amount of calculation to be performed and (iii) the size of messages to be passed in a parallel implementation, all by a factor of 2 [10].

3 Results and Conclusions

A large number of N-body force evaluations are performed using the two methods, with $N = \{1 \times 10^i, 2 \times 10^i, 5 \times 10^i; i = 2, 3, 4, 5\}$ for $p = 0$ (monopole term only), $p = 1$ (dipole moments), $p = 2$, 4, 7, 9 and 12, for $\epsilon = 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}$ and 10^{-5} respectively, where ϵ is the least upper bound to the error which is defined *a priori*, cf. eqn.(4). Care is taken to ensure that all other parameters, such as tree depth, are optimised with respect to N and ϵ. The resulting execution times were produced on Sun IPC Workstations, and from these 'wall-clock' times we are able to determine which of the two methods, the $O(N)$ or $O(N \log N)$ method, is faster for a given N and ϵ. The programs are timed from the moment after the locations and strengths of the particles have been read from file, until the time at which the final potential has been evaluated.

Two different distributions were used to compare the two methods; a uniform distribution of particles over a unit cube, and a distribution over the surface of a sphere, where the θ and ϕ of the particles' spherical coordinates are uniformly distributed over $[0, \pi]$ and $[0, 2\pi]$ respectively.

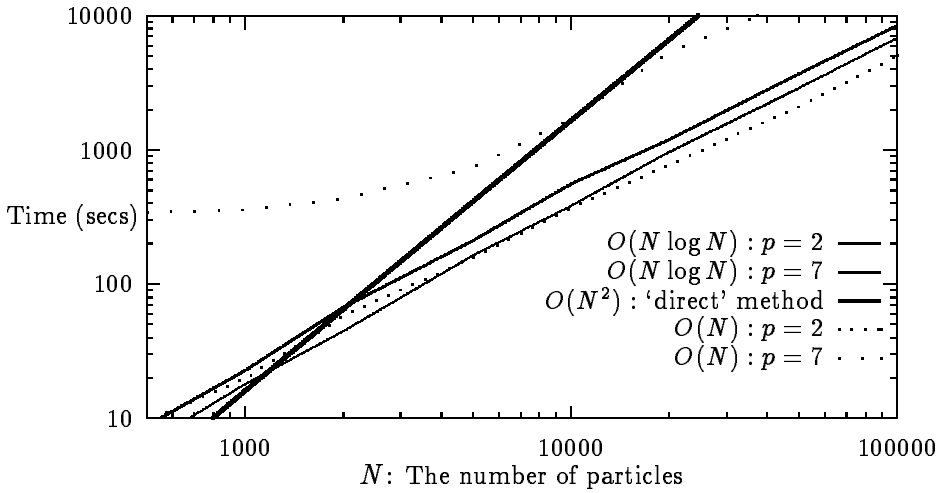

FIG. 2. *Execution times using a spherical distribution of particles.*

Figure 2 shows the execution times for the $O(N \log N)$ method for $p = 2$ and $p = 7$, the 'direct' $O(N^2)$ method and the $O(N)$ method for $p = 2$ and $p = 7$, using the spherical distribution. The execution times for the same set of parameters, but using the uniform distribution, produces a very similar graph.

For both distributions, the $O(N)$ method was substantially slower than the $O(N \log N)$

method for $N \in [10, 500K]$ and $p = 4$, 7, 9 and 12. Using the spherical distribution, with $p = 0, 1$ and 2, the $O(N)$ method became faster than the $O(N \log N)$ method when $N = 20$, 350 and 4500 respectively. When the uniform distribution was employed, with $p = 0$ and 1, the $O(N)$ method became faster than the $O(N \log N)$ method when $N = 50$ and 400 respectively. For $p = 2$, the two methods executed in approximately the same time for $N \in [10, 500K]$.

From these results we have concluded that for problems characterised by either type of distribution, the $O(N)$ method is faster for problems with low precision and large N, such as some astrophysical simulations. Whereas the $O(N \log N)$ method is more suited to problems which demand a large N and a high precision, such as the Vortex Methods in CFD, which require a high precision in order to minimise numerically induced instabilities.

3.0.1 Parallelisation. Consider a MIMD distributed memory machine, using a local domain decomposition, where the computational domain is distributed evenly over the nodes [11]. The two methods discussed in this paper require the same communication routines, and send the same data between the same nodes, i.e. the multipole moments, thus the parallel codes will only differ in a computation which is independent of inter-processor communications. Therefore this manner of parallelisation will not affect the relative performance of the two methods. However, parallelisation will allow for an extended $N - \epsilon$ space due to the larger memory resource. Moreover, to to ensure a balanced load for problems where the distribution is non-uniform, a scattered domain decomposition should be used [2, 14, 16].

References

[1] Anderson, C.R., SIAM J. Sci. Stat. Comput., **13**, 4, p923-947, July 1992.
[2] Baden, S.B., *Vortex Methods*, Lecture Notes in Mathematics, Springer Verlag, p96-119, 1988.
[3] Barnes, J., Hut, P., Nature, **324**, p446-449, 1986.
[4] Greengard, L. Gropp, W.L., Parallel Proc. for Sci. Comp., SIAM, p213-222, 1989.
[5] Greengard, L., Rokhlin, V., J. Comp. Phys., **73**, p325-348, 1987.
[6] Greengard, L., *The Rapid Evaluation of Potential Fields in Particle Systems*, MIT Press, 1988.
[7] Leathrum, J.F., Board, J.A., *The Parallel Fast Multipole Algorithm in Three Dimensions*, Technical Report, Duke University, April 1992.
[8] Lustig, S.R., Cristy, J.J., Pensak, D.A., Materials Research Society Symposium Proceedings Series, **278**, Symp. on Comp. Methods in Mat. Sci., April 1992.
[9] Nyland, L.S., Prins, J.F., Reif, J.H., 2nd Symposium on Issues and Obstacles in the Practical Implementation of Parallel Algorithms and the Use of Parallel Machines (DAGS'93), Hanover, N.H., June 1993.
[10] Pringle, G.J., Ph.D. Thesis, Napier University, Edinburgh, 1994.
[11] ——, J. Future Generation of Computing Systems, **104**, Jan., 1995.
[12] ——, *User Guide to the 3-dimensional Fast Multipole Method*, Technical Report, Napier University, Edinburgh, 1994.
[13] ——, *Error Analysis of the Multipole Methods*, Technical Report, Napier University, Edinburgh, 1994.
[14] Salmon, J.K., Warren, M.S., Winckelmans, G.S., Intl. J. Supercompter Appl., **8**, 2, 1994, (to appear).
[15] Schmidt, K.E., Lee, M.A., J. of Stat. Phys, **63**, Nos. 5/6, 1991.
[16] Singh, J.P., Ph.D. Thesis, Stanford University, 1993.
[17] Zhao, F., Johnson, S.L., J. Sci. Stat. Comput., **12**, 6, Nov. 1991.