JOURNAL OF COMPUTATIONAL PHYSICS 88, 393–408 (1990)

Comparison of Two Different Tree Algorithms

JUNICHIRO MAKINO

Department of Earth Sciences and Astronomy, College of Arts and Sciences, University of Tokyo, 3-8-1 Komaba, Meguro-ku, Tokyo 153, Japan

Received January 24, 1989

The efficiency of two different algorithms of hierarchical force calculation is discussed. Both algorithms utilize the tree structure to reduce the cost of the force calculation from $O(N^2)$ to $O(N \log N)$. The only difference lies in the method of the construction of the tree. One algorithm uses the oct-tree, which is the recursive division of a cube into eight subcubes. The other method makes the tree by repeatedly replacing a mutually nearest pair in the system by a super-particle. Numerical experiments showed that the cost of the force calculation using these two schemes is quite similar for the same relative accuracy of the obtained force. The construction of the mutual-nearest-neighbor tree is more expensive than the construction of the oct-tree roughly by a factor of 10. On the conventional mainframes this difference is not important because the cost of the tree construction is only a small fraction of the total calculation cost. On vector processors, the oct-tree scheme is currently faster because the tree construction is relatively more expensive on the vector processors.

1. INTRODUCTION

The most time-consuming part of an N-body code is the force calculation. A naive approach requires $O(N^2)$ computing cost, which makes the simulation with $N > 10^4$ rather unpractical even on the fastest supercomputers. Recently, several "tree" algorithms have been developed which map a hierarchical tree structure on an N-body system (Appel [1], Jernigan [2], Porter [3], Press [4], Barnes and Hut [5], henceforth BH, Benz, Bowers, Cameron, and Press [6], henceforth BBCP). In these algorithms, the force from a group of distant particles is approximated by a multipole expansion; typically an expansion up to the quadrupole order is used. Tree structures provide a systematic way of performing this approximation, thereby reducing the cost of the force calculation from $O(N^2)$ to $O(N \log N)$. The important advantage of these tree schemes over other fast schemes such as particle–mesh or particle–particle particle–mesh schemes is that tree schemes are gridless, so their spatial resolution is not limited by the mesh size. In addition, unlike the P^3M scheme, the efficiency of tree schemes shows only weak dependence on the spatial structure of the system.

Two very different approachs have been used to construct the tree. In the BH scheme an oct-tree is used. An oct-tree is formed by recursively subdividing a cube. On the other hand, most other approachs rely on forming the tree by creating

0021-9991/90 $3.00

"clumps." In BBCP, the mutually nearest pairs are repeatedly replaced by nodes, until there remains only one node.

The BH algorithm has been extensively analyzed both numerically and theoretically [7, 8]. No such study on the nearest-neighbor schemes has been published yet.

Here we will present an empirical comparison of these two schemes. In Section 2 we will briefly describe these two schemes. For more details see [5–7]. In Section 3 we will give the result of the comparison of these two schemes for various parameter choices. We mostly discuss spherically symmetric systems. The tree construction for the BBCP scheme is a factor of order 10 more expensive than that of the BH tree. However, this difference is not important on conventional mainframes or workstations since the cost of the tree construction is trivial for either algorithm. The error and efficiency measurements obtained for BH scheme are in close agreement with those of [7, 8]. The relative accuracy of the BH and the BBCP force calculation algorithms are quite similar.

In Section 4 we discuss the performance of these algorithms on vector/parallel computers. The BH scheme is proven to be efficient on a wide variety of machines [9–13]. The force calculation using the BBCP tree can be vectorized/parallelized in the same way as the BH scheme; we are actually able to use the same code for the force calculation in both schemes. Thus the efficiency and cost of vectorized calculations is similar for both schemes, except for the small difference caused by the difference in the tree geometry. However, the construction of the BBCP tree has not been effectively vectorized yet. This difference results in a fairly large difference in the CPU time. A vectorizable algorithm for the nearest neighbor search might be able to reduce the difference in the CPU time.

2. Description of the Algorithms

2.1. *The BH Tree Construction*

Tree construction in the BH scheme is quite simple. First we create a cube that is large enough to contain all particles in the system. Then we recursively subdivide the cube into eight sub-cubes, until each cube contains one or no particle. Figure 1 shows a BH tree for a two-dimensional case. The largest cube corresponds to the root node of the tree and sub-cubes of a cube form child nodes of the node corresponding to that cube. Barnes and Hut [5, 8] originally used a very different scheme to construct the tree. Their scheme is somewhat faster and requires less memory, at least on a conventional mainframe. However, their scheme is difficult to vectorize and our scheme is vectorizable [12]. Moreover, present supercomputers have very large amounts of memory—or they are far too slow for calculations large enough to consume all of their memory—and the difference in the memory requirement is of little importance.

On conventional mainframes or workstations, the cost of the BH tree construc-

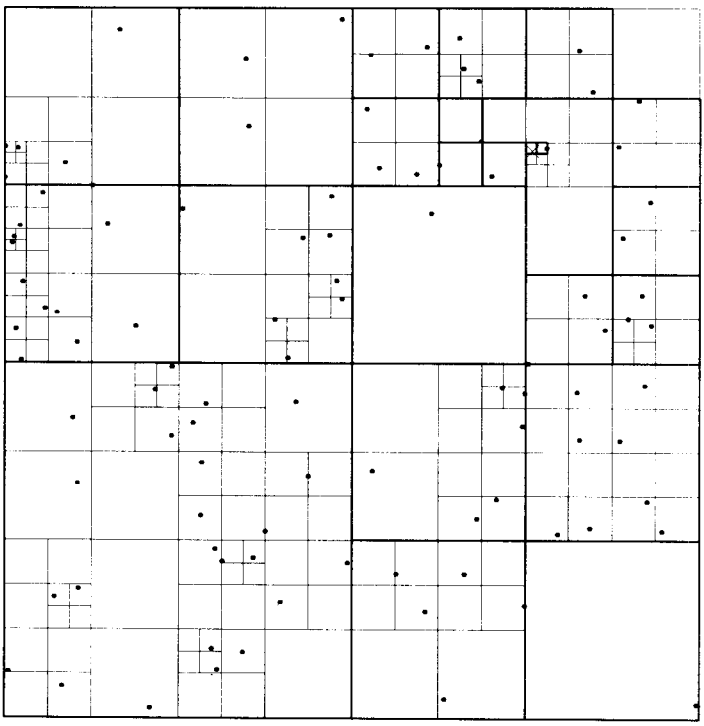

FIG. 1. The quad-tree for the random distribution of 100 particles in the square of the unit size.

tion is less than 5% of that of the force calculation, even for the relatively low required accuracy. For the accuracy of practical levels, the cost of the BH tree construction is less than 2%.

2.2. *The BBCP Tree Construction*

In the BBCP scheme, the tree is constructed in a bottom-up order, by replacing the mutually nearest pair with a node. The algorithm is schematically expressed as follows:

```
subroutine maketree ( )
initialize the list of active particles list so that it contains all particles
while (the list contains more than one particle)
  foreach i in list
    nnb(i) = nearest neighbor of i
  endforeach
  clear the list of mutual nearest pairs pairlist
```

```
    foreach i in list
      if (nnb(nnb(i)) eq. i) then
        append pair i, nnb(i) to pairlist
      endif
    endforeach
    foreach (i, j) in pairlist
      remove i and j from list
      create the parent node for i and j
      append the parent node to list
    endforeach
  endwhile
```

Figure 2 shows a BBCP tree for the two-dimensional case. The most expensive part of this algorithm is the search for the nearest neighbor. BBCP gives a fairly efficient scheme to find nearest neighbors for all particles in an N-body system, with the cost of order $O(C_1 N + C_2 N \log N)$. We used this scheme for all calculations in this paper. It should be noted that this scheme is not quite optimized yet. For example, the monotonic logical grid scheme [14] may show better performance in the nearest neighbor search.

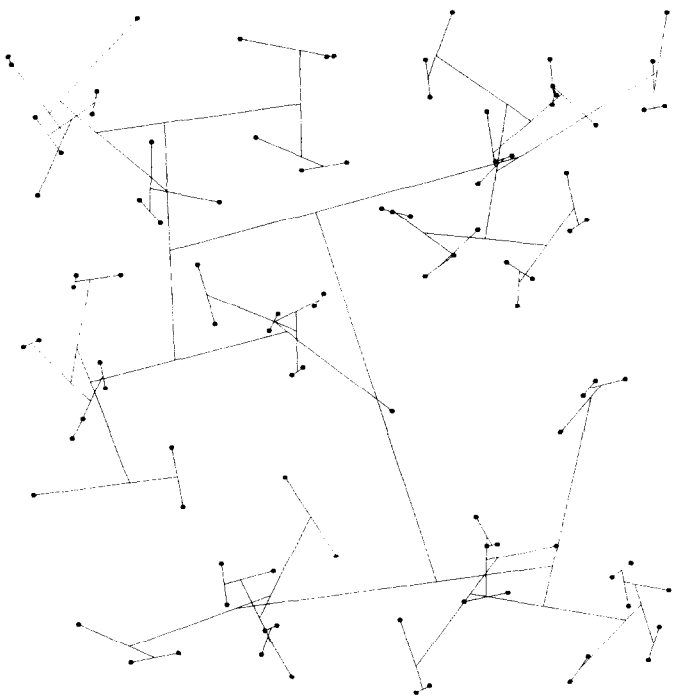

FIG. 2. The BBCP-tree for the random distribution of 100 particles in the square of the unit size.

In the current implementation, the tree construction takes about 40 s for a 2048 particle Plummer model on a Sun 3/60 with 68881. The force calculation takes about 150 s on the same machine for an accuracy of $\sim 1\%$. Thus on workstations such as SUNs, the cost of tree construction is hardly crucial.

2.3. *The Force Calculation*

The basic algorithm for the force calculations is the same for both types of trees and is expressed as follows:

```
subroutine treeforce (i, node, force)
  if (node and particle i are well separated)
    force = force from the total mass in the center of mass of node
  else
    force = 0
    do j = 1, number_of_children(node)
      call treeforce(i, child(node, j), child_force)
      force = force + child_force
    enddo
  endif
  return
```

The total force on particle **i** is obtained by calling **treeforce(i, root, force)**. The condition to determine if a node is well separated from a particle is

$$\begin{aligned} \frac{l}{d} &< \theta \qquad \text{(BH)} \\ \frac{R}{d} &< \theta \qquad \text{(BBCP)}, \end{aligned} \tag{1}$$

where l is the size of the cube in the BH tree corresponding to the node, R is the "radius" of the cell defined in Eq. (A8) of BBCP, d is the distance between the center of the mass of the node and the particle, θ is the parameter which controls the accuracy as well as the calculation cost. These criteria are not optimal and it is possible to improve the accuracy without increasing the calculation cost (see Appendix B). However, the loss of efficiency is not large; less than a factor of two. Therefore we adopted the above original criterion in this paper.

This algorithm is recursive and not well suited for the implementation by Fortran 77 or vectorization. For different modifications of this basic algorithm see [10–12]. It is interesting that all these three schemes vectorize different aspects of the force calculation, and therefor can be applied simultaneously. For a shared time step integrator in which the forces on all particles are calculated at each timestep, the combination of the Makino's method [12] and Barnes's method [10] will

probably show the best performance. For a individual time step scheme, the combination of Makino's method [12] and Hernquist's method [11] is relatively easy to implement and would probably show reasonable performance.

3. Comparison of the Efficiency

3.1. *Comparison of the Amount of Computation*

Figure 3 shows the error in the force as the function of the accuracy parameter θ for both BH and BBCP trees. Random realizations of the Plummer model are used for the N-body system. The error is defined as

$$e = \frac{|\mathbf{F}_{\text{tree}} - \mathbf{F}_{\text{direct}}|}{|\mathbf{F}_{\text{direct}}|}, \tag{2}$$

where $\mathbf{F}_{\text{tree}}$ is the force obtained by the tree algorithm, $\mathbf{F}_{\text{direct}}$ is that obtained by the $O(N^2)$ direct summation. 64-bit arithmetic is used throughout the experiment. We plot the r.m.s. error

$$\langle e \rangle = \left(\frac{1}{N} \sum_{i=1}^{N} e_i^2 \right)^{1/2}, \tag{3}$$

where N is the total number of the particles in the system and e_i is the error in the force on particle i. We showed the error of the potential and the force, for both of the monopole and quadrupole calculations.

The BH tree is more accurate than the BBCP tree for the same value of θ. However, this difference is mainly because of the difference in the definition of the accuracy parameter. For BH scheme, we used the length of the one side of the cube. For BBCP scheme, we used the radius of the node. Thus, there is a difference of roughly a factor of 2 in the representative size used in these schemes.

Figure 4 shows the error as the function of the average number of force terms evaluated in the force calculation N_{terms}. Here, the BBCP scheme shows a comparable accuracy for the same number of terms. To be precise, the BBCP tree gives significantly more accurate values for the potential, especially with quadrupole correction. However, there is no such clear difference in the obtained gravitational force. It seems that the BBCP tree tends to be more accurate than the BH tree when N_{terms}/N is large. With practical number of N_{terms}, however, the difference in the accuracy between these two schemes is small. When only the monopole term is used, the BH tree tends to be more accurate.

It is difficult to explain these differences because there is no simple theory about the behavior of the BBCP tree. In Appendix A we give a simple model for the behavior of the calculation cost and the error in the case of the BH tree.

The difference in the obtained accuracy is not so large as to be the decisive factor

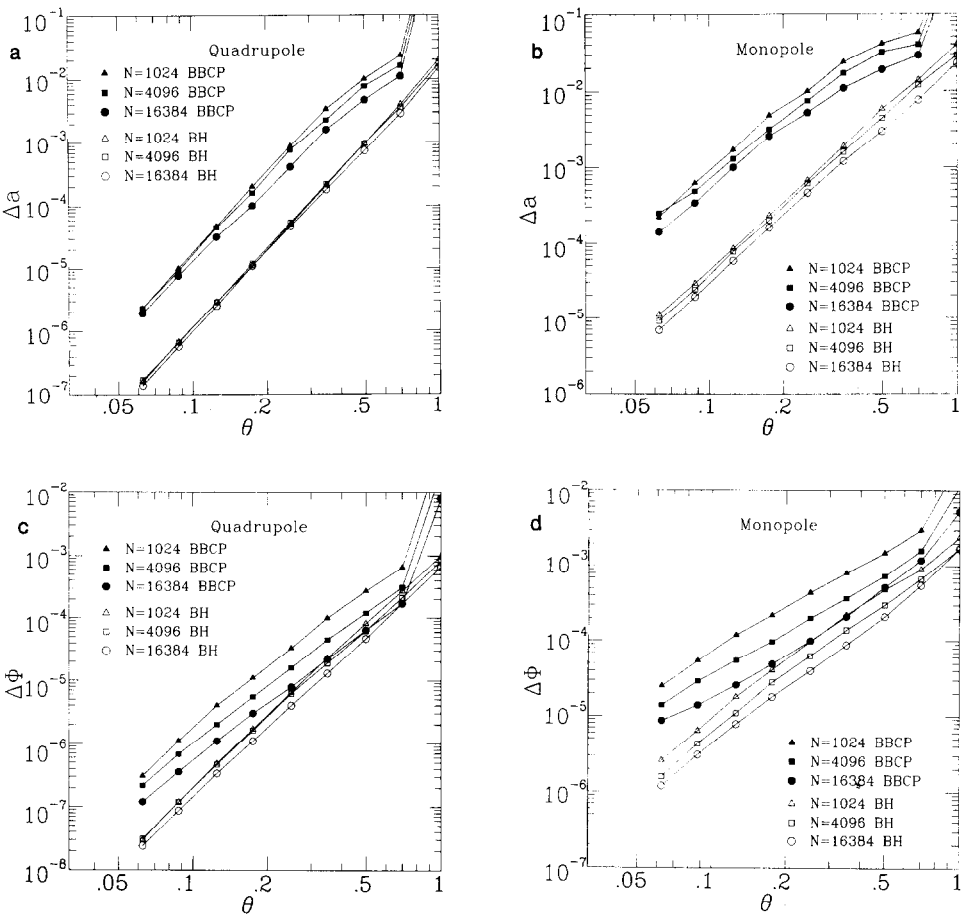

FIG. 3. The r.m.s. error of the gravitational force and potential as a function of the accuracy parameter θ. Open symbols indicate the error of the BH tree and filled symbols indicate that of the BBCP tree. Triangles are for $N = 1024$, squares are for $N = 4096$, circles are for $N = 16384$. The Plummer model is used as the particle distribution: (a) force error with quadrupole term; (b) force error with monopole; (c) potential error with quadrupole, (d) potential error with monopole.

to prefer one algorithm over the other, because other factors, especially the efficiency in the vector pipeline, can easily be more important. In the following subsections we will discuss the actual cost in some detail.

3.2. *Timing Results*

Here we give the result of the CPU time comparison between the BH tree and the BBCP tree. We used a Plummer model with $N = 8192$ as the test N-body system on the scalar mainframe and $N = 16384$ on the vector processor. Table I shows the average CPU time per one force calculation on a FACOM M-360 mainframe and

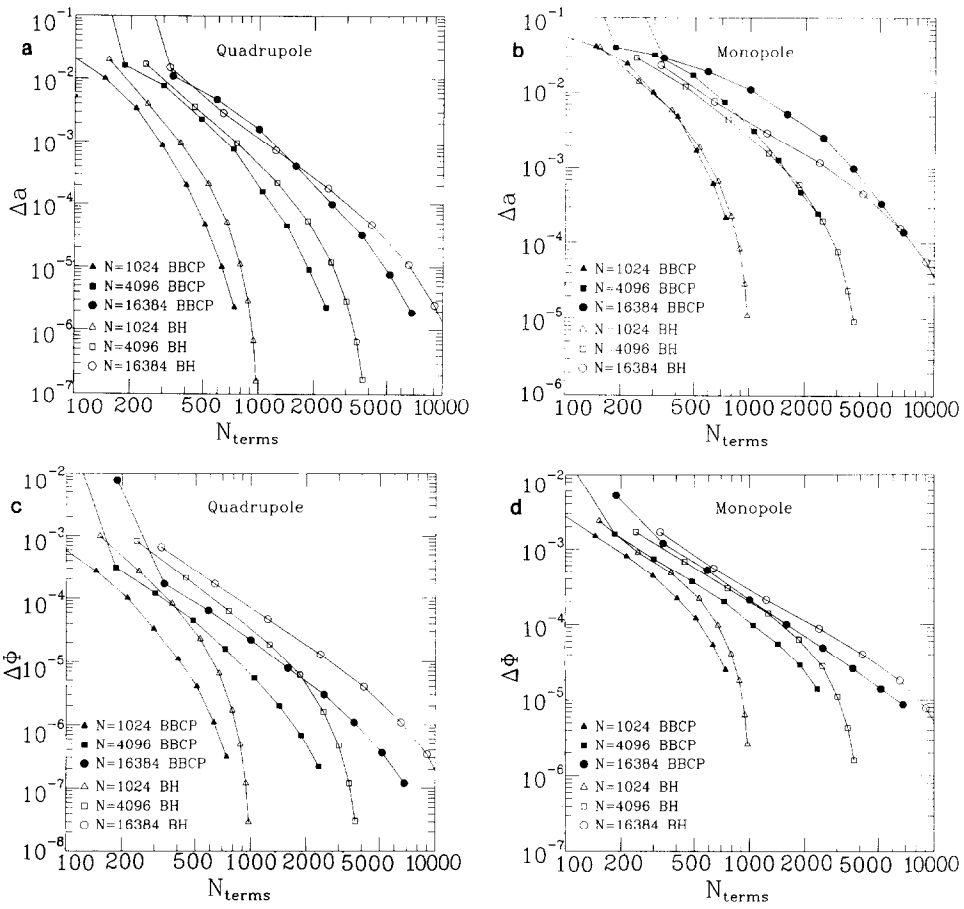

FIG. 4. Same as Fig. 3 but plotted as the function of the average number of force terms evaluated in the force calculation.

a FACOM VP-400 vector processor. On both machine the code based on Makino's [12] vectorizing scheme is used. Therefore the timing on scalar machine may not indicate its best performance.

It is not surprising that the BBCP scheme takes considerably longer time. The reason is that the BBCP tree is usually deeper than the BH tree, because the former

TABLE I

CPU Time per Interaction

	Scalar (M-360)	Vector (VP-400)
BH	100μs	0.82μs
BBCP	150μs	1.25μs

TABLE II

CPU Time for the Tree Construction

	Scalar (M-360, $N = 8192$)	Vector (VP-400, $N = 16384$)
BH	2.4 s	0.39 s
BBCP	38 s	22.1 s
Force (BH)	230 s	4.39 s

is the binary tree and the latter is the oct-tree. Therefore, the number of nodes to be opened up in the BBCP tree is significantly larger than that in the BH tree. To be precise, the number of nodes that are opened up is always equal to $N_{\text{terms}} - 1$ in the BBCP tree. In the BH tree, the number of nodes to be opened up depends on the distribution of particles in the system, because the BH tree is not the spanned tree. Experiments indicate that it is about $0.2 N_{\text{terms}}$. This difference is not negligible either on scalar processors or on vector/parallel processors.

Table II shows the time required for the tree construction. The time for the force calculation to the average accuracy of about 1% ($\theta = 1$, BH tree) is also shown for comparison. The cost of the BBCP tree construction is significantly larger than that for the BH tree. We should note that there is relatively large room for improvement in the tree construction algorithm of BBCP tree. Roughly 90% of the CPU time is spent in the nearest neighbor search. In the current implementation the tree is constructed from scratch at each time step, and the nearest neighbors are searched at each iteration of the tree construction algorithm. It should be possible to recycle the old nearest neighbors in some way. Moreover, it is not necessary to reconstruct the tree each times. Typically the reconstruction of the tree is necessary for only once in ten or more time steps [15]. Thus the cost of the tree construction can be reduced by a factor larger than ten. Thus at least on conventional scalar machines, the cost of the tree construction is negligible even for the BBCP tree, unless we use the individual time steps (e.g., Hernquist and Katz [16]).

4. Summary

In this paper we discussed the efficiency of the two different tree algorithms. The conclusion is that there is no significant difference in the calculation cost, except that the construction of the BBCP tree is currently more expensive. Intuitively, force calculation using the BBCP tree would seem to be more efficient than force calculation using the BH tree, because the former tree appeal to more naturally reflect the clustering in the system as suggested in [4, 6]. However, this effect turned out to be relatively small compared to other factors such as the difference in the algorithmic efficiency of the tree construction and the vectorizability of the algorithm.

APPENDIX A: THE SCALING OF THE CALCULATION COST AND THE ERROR

Here we give an intuitive explanation of the scaling law for the average error and computing cost of the BH-tree. The behavior of the BBCP tree is more difficult to analyze rigorously and not discussed here. We consider a system of N particles randomly distributed in a cube of the unit size. With a realistic particle distribution the theoretical treatment becomes much more difficult and we will not try it here. To simplify the discussion, here we discuss the number of terms and the error evaluated at the center of the cube. We assume

$$N = 8^n, \tag{A1}$$

and that the tree is strictly uniform, i.e., the tree is the spanned tree of the depth n. In practice, the tree is not strictly uniform even in the case of the uniform particle distribution, because of local fluctuations. For example, average distance between the components of the closest pair in a N-body system is proportional to $N^{-2/3}$, implying that the deepest level of the tree is $2\log_8 N$ instead of $\log_8 N$.

It is relatively straightforward to include the effect of local fluctuations strictly. However, it is rather tedious and does not change the conclusion at least when $\theta > N^{-1/3}$. Thus here we neglect local fluctuations.

The average number of nodes in the level l which requires subdivision according to the criterion (1) is

$$N_{\mathrm{div},l} = \begin{cases} \dfrac{4\pi}{3}\theta^{-3}, & (0 > 2^{-l+1}), \\ g(l,\theta)\dfrac{4\pi}{3}\theta^{-3}, & (2^{-l} < \theta < 2^{-l+1}), \\ 0, & (\theta < 2^{-l}), \end{cases} \tag{A2}$$

where $g(l, \theta)$ is a positive function that does not exceed 1. The number of the nodes in level l on which we apply the multipole approximation is obtained by subtracting the number of the node in level l that are subdivided from the number of the children of the divided nodes in level $l-1$, that is,

$$\begin{aligned} N_{\mathrm{term},l} &= 8N_{\mathrm{div},l-1} - N_{\mathrm{div},l} \\ &\simeq \begin{cases} \dfrac{28\pi}{3}\theta^{-3}, & (\theta > 2^{-l+2}), \\ \dfrac{28\pi}{3}\theta^{-3} g(l-1,\theta), & (2^{-l+1} < \theta < 2^{-l+2}), \\ 0, & (\theta < 2^{-l+1}). \end{cases} \end{aligned} \tag{A3}$$

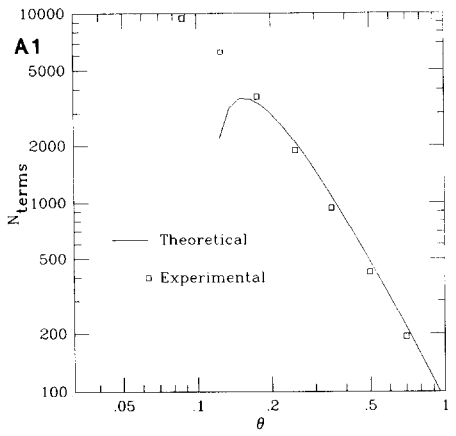

FIG. A1. The number of the force terms per one force calculation plotted as a function of the accuracy parameter θ. The particle distribution is the uniform distribution within a cube. The number of particles in the system is 16384. The solid line indicates the theoretical estimate and the open squares indicate the experimental value.

The total number of the terms for the force calculation is thus

$$N_{\text{terms}} = \sum_{l=0}^{n} N_{\text{term},l}$$
$$\simeq 28(n - 1.5 + \log_2 \theta)\,\theta^{-3}$$
$$\simeq 31\theta^{-3} \log_{10} \frac{N\theta^3}{23}. \tag{A4}$$

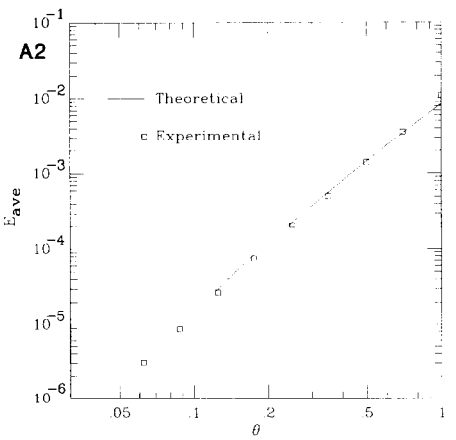

FIG. A2. Same as Fig. A1 but the r.m.s. error of the obtained force is plotted.

To obtain Eq. (A4), we assume

$$g(l, \theta) = \text{const} = 0.5. \tag{A5}$$

Equation (A4) cannot be applied when $\theta \lesssim 2^{-n+1.5}$. Figure A1 shows the number of terms for a 16384-body system with a uniform random distribution of particles in a cube. Solid line indicate the theoretical estimate of (A4) and circles indicate the measured number of terms averaged over all particles. The theoretical estimate shows reasonably good agreement with the measured value in the range $\theta > 0.2$.

The leading term of the force error comes from the lowest order term of the multipole expansion truncated. As discussed in [8] in detail, with the BH tree the average magnitude of this lowest term is proportional to square root of the number of the particles in the cell when the system can be regarded as homogenous, because the quadrupole and the octapole moments vanishes in the continuous limit. Thus, if the system is homogeneous, the magnitude of the error of the force on a particle from a node in the case of the center-of-mass approximation can be estimated as

$$e \propto \theta^2 d^{-2} N_p^{1/2}, \tag{A6}$$

where θ is the accuracy parameter defined in Eq. (1), d is the distance between the particle and the node, N_p is the number of particles in the node. Here we assume that all particles have an equal mass. From Eq. (1), it is clear that the average distance of the nodes in level l is proportional to the size of the box. Thus we obtain

$$d \propto \frac{2^{-l}}{\theta}. \tag{A7}$$

The average number of particles in a cube is proportional to the volume of the cube, i.e.,

$$N_p \propto 2^{-3l}. \tag{A8}$$

By applying (A7) and (A8) into (A6), we obtain the average error from one node as the function of the level of the node l and the accuracy parameter θ,

$$e(l, \theta) = C\theta^4 2^{l/2}. \tag{A9}$$

Now we can estimate the total error, which is expressed as

$$\begin{aligned} \langle e^2 \rangle^{1/2} &= \left\{ \sum_{l=0}^{n} N_{\text{term},l} [e(l, \theta)]^2 \right\}^{1/2} \\ &\simeq C_1 \theta^{5/2} \left(\frac{N^{1/3}}{8} - \frac{1}{\theta} \right)^{1/2}, \end{aligned} \tag{A10}$$

where C_1 is a constant. When the quadrupole moment is retained, (A6) is replaced by

$$e \propto \theta^3 d^{-2} N_p^{1/2}. \tag{A11}$$

and the total error is expressed as

$$\langle e^2 \rangle^{1/2} \simeq C_2 \theta^{7/2} \left(\frac{N^{1/3}}{8} - \frac{1}{\theta} \right)^{1/2}. \tag{A12}$$

Figure A2 is shows the experimental error and the theoretical one for the same N-body system as is used in Fig. A1. The value of C_1 is chosen so that the theoretical estimate is close to the experimental result. These two show quite good agreement.

APPENDIX B: MORE EFFICIENT OPENING CRITERION

In this paper we used the opening criterion of the form

$$d > \frac{l}{\theta}, \tag{B1}$$

where d is the distance between the node and the particle, l is the representative size of the node, and θ is a constant parameter. The above inequality is a conservative criterion in the sense that it assures that the local error will not exceed an certain upper bound. However, it is not necessarily an optimal one in view of the computational cost. The criterion B1 poses the upper bound on the local relative error. On the other hand, to minimize the calculation cost to obtain a fixed accuracy, we should control the absolute local error of each force term to the same order. The average absolute error of the force from one node is given in Eq. (A6). By eliminating θ we obtain

$$e = \begin{cases} A l^2 d^{-4} N_p^{1/2} & \text{(monopole)} \\ B l^3 d^{-5} N_p^{1/2} & \text{(quadrupole)}, \end{cases} \tag{B2}$$

where A and B are some constants. Thus, the opening criterion to keep the average local error to a constant e_{lim} is

$$\begin{aligned} d &> (A e_{\text{lim}})^{-1/4} N_p^{1/8} l^{1/2} && \text{(monopole)} \\ d &> (B e_{\text{lim}})^{-1/5} N_p^{1/10} l^{3/5}; && \text{(quadrupole).} \end{aligned} \tag{B3}$$

Figure B1 shows the average error in the force as a function of the number of force terms for the original criterion (B1) and the modified criterion (B3). The BH tree on a 16384-body Plummer model is used. The modified opening criterion gives the average error smaller than that of original one by a factor two or more.

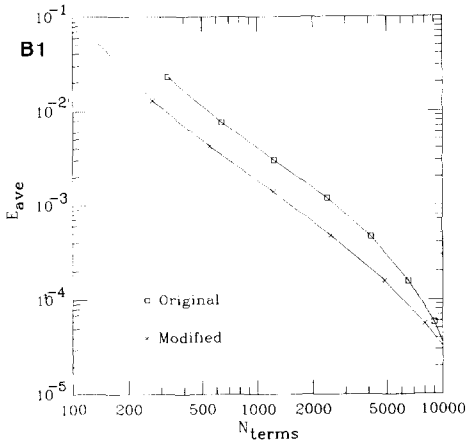

FIG. B1. The r.m.s. error of the force as the function of the number of force terms in the case of the BH tree. Open squares indicates the error for the force obtained with original opening criterion. Crosses are for the modified criterion (B3). The Plummer model with $N = 16384$ is used.

In the homogeneous limit, the number of particles in a cell is proportional to its volume, i.e.,

$$N_p \propto l^3. \tag{B4}$$

From (B3) and (B4) we obtain

$$d \propto \begin{cases} l^{7/8} & \text{(monopole)} \\ l^{9/10} & \text{(quadrupole)}, \end{cases} \tag{B5}$$

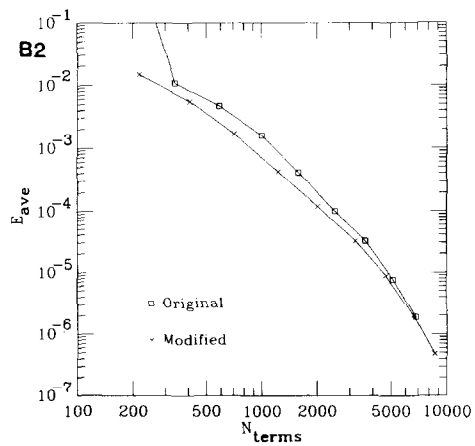

FIG. B2. Same as Fig. B1 but for the BBCP tree and criterion (B8).

or, by introducing the opening criterion $\theta(l) = l/d$,

$$\theta(d) = \begin{cases} \theta_0 l^{1/8} & \text{(monopole)} \\ \theta_0 l^{1/10} & \text{(quadrupole)}, \end{cases} \tag{B6}$$

where θ_0 is the opening angle at $d = 1$. Equation (B6) implies that the opening angle for the nearby nodes is effectively smaller than that for the far nodes in the case of the modified criterion. The reason why nearby nodes requires smaller opening angle is that the average error from nearby node is larger than that from far nodes, when constant opening angle is used, as shown in (A9).

In practice the criterion (B3) is rather dangerous and should be used in combination with the original criterion. The reason is that the convergence of the multipole expansion is not guaranteed for large cells, since the criterion (B3) leads to a too-large opening criterion for large cells. It should be also noted that, strictly speaking, (B2) holds only if the distribution of particles within the cell is homogeneous (see, however, [8]).

In the case of the BBCP tree, it is not easy to derive a theoretically optimal opening criterion. Originally BBCP used the criterion (B1) with l recursively defined as

$$l = \max(l_1 + r_1, l_2 + r_2), \tag{B7}$$

where l_i is the size of the child i and r_i is the distance between the center of the mass of the node and that of the child i. The size of a particle is defined as 0. This criterion is, however, clearly not optimal. Nodes in the lowest level that contain only two particles always have the maximum possible quadrupole moment measured in the unit of l^2. Therefore, it seems desirable to apply somewhat modified "size," so that the size of the lower nodes are effectively larger than that determined by (B7). We found that the following modified "size" gives more accurate force at the same calculation cost.

$$l = \max(l_{\text{BBCP}}, kr_1\ kr_2), \tag{B8}$$

where l_{BBCP} is the size of the node defined by (B7), k is a constant parameter larger than 1. Figure B2 shows the error as the function of the calculation cost for both the original and modified criterion with $k = 1.5$. This value is chosen as one example and we have not yet investigated the optimal value for k. The modified criterion gives the accuracy roughly factor of two better than the original criterion.

ACKNOWLEDGMENTS

I would like to thank Joshua Barnes and Lars Hernquist for providing their code and for valuable comments on the manuscript, Piet Hut for useful and stimulating discussions, William Press for comments on the manuscript. Computations were perfomed at The Educational Computer Center of University of Tokyo and Institute for Supercomputing Research, Tokyo, Japan.

REFERENCES

1. A. APPEL, *SIAM J. Sci. Stat. Comput.* **6**, 85 (1985).
2. G. JERNIGAN, in *Dynamics of Star Clusters, I.A.U. Symp.* 113, edited by J. Goodman and P. Hut (Reidel, Dordrecht, 1986), p. 275.
3. D. PORTER, Ph.D. thesis, Physics Department, University of California, Berkeley, 1985 (unpublished).
4. W. PRESS, in *The Use of Supercomputers in Stellar Dynamics*, edited by S. L. W. McMillan and P. Hut (Springer-Verlag, Berlin, 1986), p. 184.
5. J. BARNES AND P. HUT, *Nature* **324**, 446 (1986).
6. W. BENZ, R. L. BOWERS, A. G. W. CAMERON, AND W. H. PRESS, preprint (1988).
7. L. HERNQUIST, *Ap. J. Suppl.* **76**, 64 (1987).
8. J. BARNES AND P. HUT, *Ap. J. Suppl.* **70**, 389 (1989).
9. J. BARNES, in *The Use of Supercomputers in Stellar Dynamics*, edited by S. L. W. McMillan and P. Hut (Springer-Verlag, Berlin, 1986), p. 175.
10. J. BARNES, *J. Comput. Phys.* **87**, 161 (1990).
11. L. HERNQUIST, *J. Comput. Phys.* **87**, 137 (1990).
12. J. MAKINO, *J. Comput. Phys.* **87**, 148 (1990).
13. J. MAKINO AND P. HUT, *Comput. Phys. Rep.* **9**, 199 (1989).
14. J. P. BORIS AND S. G. LAMBRAKOS, in *Proceedings, State-of-the-Art, Free-Lagrangian Methods Meetings*, 1985, edited by M. J. Fritts *et al.* (Springer-Verlag, New York, 1985), p. 185.
15. W. H. PRESS, Department of Physics, Harvard University, private communication (1988).
16. L. HERNQUIST AND N. KATZ, *Ap. J. Suppl.* **76**, 64 (1989).