MANY-BODY TREE METHODS IN PHYSICS

SUSANNE PFALZNER
Max-Planck-Research Unit
"Dust in Starforming Regions,"
University of Jena

PAUL GIBBON
Max-Planck-Research Unit
"X-Ray Optics,"
University of Jena

CAMBRIDGE
UNIVERSITY PRESS

Published by the Press Syndicate of the University of Cambridge
The Pitt Building, Trumpington Street, Cambridge CB2 1RP
40 West 20th Street, New York, NY 10011-4211, USA
10 Stamford Road, Oakleigh, Melbourne 3166, Australia

First published 1996

Printed in the United States of America

Library of Congress Cataloging-in-Publication Data
Pfalzner, Susanne.
Many-body tree methods in physics / Susanne Pfalzner, Paul Gibbon.
p. cm.
Includes bibliographical references and index.
ISBN 0-521-49564-4 (hardcover)
1. Many-body problem. 2. Algorithms. 3. Mathematical physics.
I. Gibbon, Paul. II. Title
QC174.17.P7P44 1996 95-36237
530.1′44 – dc20 CIP

A catalog record for this book is available from the British Library.

ISBN 0-521-49564-4 hardback

MANY-BODY TREE METHODS IN PHYSICS

To our daughter, Theresa,
and our parents, Helga and Hans and Elin and David

Preface

The difficulty in writing a 'how-to' book on numerical methods is to find a form which is accessible to people from various scientific backgrounds. When we started this project, hierarchical N-body techniques were deemed to be 'too new' for a book. On the other hand, a few minutes browsing in the References will reveal that the scientific output arising from the original papers of Barnes and Hut (1986) and Greengard and Rohklin (1987) is impressive but largely confined to two or three specialist fields. To us, this suggests that it is about time these techniques became better known in other fields where N-body problems thrive, not least in our own field of computational plasma physics. This book is therefore an attempt to gather everything hierarchical under one roof, and then to indicate how and where tree methods might be used in the reader's own research field. Inevitably, this has resulted in something of a pot-pourri of techniques and applications, but we hope there is enough here to satisfy the beginners and connoisseurs alike.

Contents

Preface *page* ix

1 Introduction 1

2 Basic Principles of the Hierarchical Tree Method 9

2.1 Tree Construction 9

2.2 Force Calculation 18

2.3 Multipole Expansion 26

2.4 Dynamics 35

3 Open Boundary Problems 37

3.1 Gravitational Problems in Astrophysics 37

3.2 Space-Charge Dominated Particle Beams 47

3.3 Collisions of Heavy Nuclei: A Case Study 52

4 Optimisation of Hierarchical Tree Codes 65

4.1 Individual Timesteps 65

4.2 Higher Order Integration Schemes 70

4.3 Vectorisation and Parallelisation 73

4.4 Timing 81

4.5 Accuracy 83

4.6 Special Hardware 87

5 Periodic Boundary Conditions 88

5.1 Minimum Image Method 90

5.2 Ewald Summation 93

5.3 Timing 102

5.4 Monte Carlo Application 104

5.5 Nonequilibrium Systems 106

6 Periodic Boundary Problems 109

6.1 Plasma Physics: Collisions in Dense Plasmas 110

6.2 Systems of More Complex Interaction Potentials 117

6.3 Biomolecules 120
6.4 Materials Science 124
7 The Fast Multipole Method 126
7.1 Outline of the Fast Multipole Algorithm 126
7.2 2D Expansion 134
7.3 3D Expansion 139
7.4 Implementation of Fast Multipole Codes 140
7.5 Timing and Accuracy 143
7.6 Applications 146
Appendix 1: Multipole Expansion in Two Dimensions 149
Appendix 2: Spherical Harmonics 152
Appendix 3: Near-Neighbour Search 155
References 157
Index 165

1
Introduction

Classical systems that consist of many particles interacting through long-range forces have interested physicists for centuries. The equation of motion of a system with more than two particles does not have an analytical solution, and it is only since the advent of high-speed computers that the trajectories of many particles could be followed simultaneously in detail. Over the last 40 years, computer simulations of N-body systems have become an indispensable tool in all branches of physics. Mathematically the N-body problem is represented by the solution of N second-order differential equations of the form

$$m_i \frac{d^2\mathbf{r}_i}{dt^2} = -\nabla_i V \qquad i = 1, 2, 3 \ldots N, \tag{1.1}$$

where $\mathbf{r}_i$ and m_i are the positions and masses of the ith particle. The knowledge of the positions and velocities as a function of time allows the global or macroscopic properties of the system to be calculated.

The potential V in Eq. 1.1 can include different kinds of interactions – those stemming from the forces the particles exert on each other and those of external fields V_{ex} like external electric or gravitational fields. For classical systems the general form of the potential can be

$$V = V_{short} + V_{long} + V_{ex}, \tag{1.2}$$

where V_{short} is a rapidly decaying function of distance, like, for example, the Van der Waals potential in chemical physics, and V_{long} is a long-range potential like, for example, the Coulombic or gravitational potential. For a comparison of a typical short-range potential with a long-range potential see Fig. 1.1. The external field V_{ex} is a function which is usually independent of the number and relative position of the particles and is calculated separately for each particle, which leads to a computation time of the order $O(N)$. In the numerical evaluation of fields the cost of computing V_{short} is of the order $O(N)$ too, because

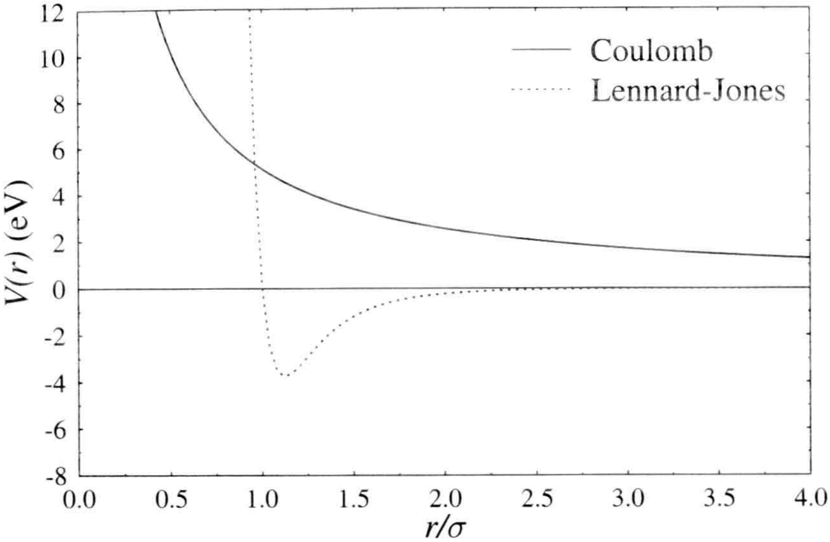

Fig. 1.1. Comparison of a typical long-range potential (Coulomb potential) with a short-range potential (Lennard–Jones potential).

the potential decays rapidly and each particle interacts significantly with only a small number of its nearest neighbours.

Although computers have made the simulation of this type of N-body problem feasible, those including long-range forces – Coulombic or gravitational – still present a challenge. The evaluation of V_{long}, if done directly, requires on the order of $O(N^2)$ operations, because this kind of potential decays slowly and the interaction of each pair of particles in the system has to be taken into account. Improvements in technique and computer speed have significantly increased the manageable simulation size, but the number of particles in such direct calculations is still too small for a variety of problems at present. This kind of computation is reasonable for a system involving a few hundred particles, but the costs increase so rapidly that simulations with a few thousands of particles are quite expensive and those with millions of particles are unattainable.

On the other hand, there are many examples in classical physics where models based on large-scale ensembles of particles interacting by long-range forces are very useful – astrophysics and plasma physics are two prominent examples. Several different approaches have been developed to reduce the burden of the long-range part of the calculation. Until recently, so-called particle-in-cell (PIC) methods have been regarded as the only effective way to simulate large systems of particles interacting through long-range forces. For a detailed review see Dawson (1983). The basic procedure of the PIC method is:

- A regular grid is laid out over the simulation region and the particles contribute their masses, charges, etc. to create a source density; this source density is interpolated at the grid points;
- the solution of the elliptical partial differential equation, usually obtained with the help of a fast Poisson solver, is used to calculate the potential values at the grid points;
- using these potential values the force is evaluated and interpolated to the particle positions.

So, by superimposing a grid of sample points, the potential field associated with the forces is calculated.

The total operation count for the PIC method is of the order $O(N + M \log M)$, where M is the number of mesh points and N is the number of particles in the system. Although the asymptotic computation cost is of the order $O(N \log N)$, in practice $M \ll N$ and, therefore, the numerical effort is observed to be proportional to N.

Due to this good computational efficiency, PIC codes are applied successfully to a variety of problems. However, there are three situations that PIC codes have difficulties dealing with:

- Strongly nonuniform particle distributions.
- Strongly correlated systems.
- Systems of complex geometry.

The first problem concerns the fact that the mesh in the standard particle-mesh schemes provides limited resolution. Due to the limitations of memory space in currently available computers, it may not be possible to use standard PIC methods to model the dynamics of a system with highly nonuniform source distributions, like, for example, in galaxies or in the cold dark matter scenario.

There have been attempts to overcome this disadvantage of the standard PIC method to obtain a better resolution. Villumsen (1989) has developed a PIC code which employs meshes of finer gridding in selected subregions of the system. Owing to the local improvement of the spatial resolution, these hierarchical meshes permit a more accurate modelling of regions of higher particle density. Where the poor resolution is of a dynamical nature – for example, due to shock waves – moving grids and adaptive grid refinement (Brackbill & Ruppel 1986) can be applied.

If the ratio of particles per cell is suitably large, local interactions are smoothed away in PIC simulations, leaving behind the collective or global behaviour of

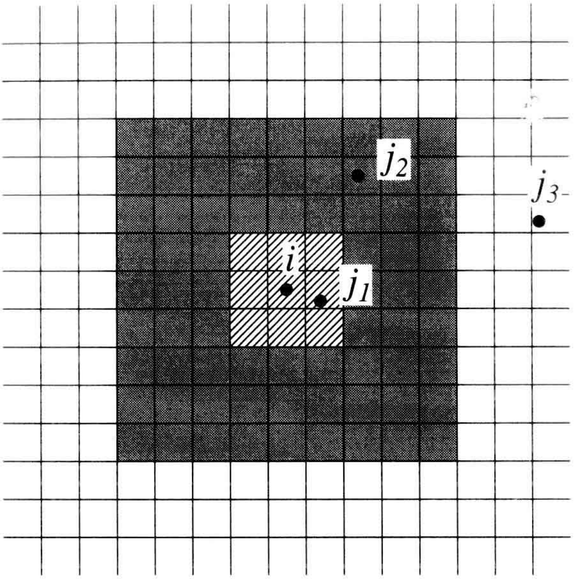

Fig. 1.2. Organisation of force contributions in P^{3}M code.

the system, and fluctuations due to the poor statistics are less of a problem. Although this smoothing effect is desirable in the context of, say, plasma kinetic theory, where the systems are usually considered uncorrelated, the PIC method is often not satisfactory where local corrections become important for the correct description of the physics. Many gravitational systems as well as high-density plasmas cannot be approximated as collisionless, and the two-body correlation function plays an important role.

A better method to describe the local correlations with PIC simulations is the so-called particle–particle particle–mesh technique (P^{3}M) (Hockney & Eastwood 1981). The idea in P^{3}M is to correct the far-field solution by including local forces by direct particle–particle interactions (see Fig. 1.2). This method seems to be an effective compromise between the possible number of particles and the spatial resolution in dynamic problems.

P^{3}M performs very well if the particles are more or less uniformly distributed in a rectangular region and relatively low precision is required. However, if the required precision is high or the particle distribution is clustered, this algorithm is sometimes not ideal. In these situations, inaccuracies can be introduced by matching long- and short-range forces (Bouchet & Hernquist 1988), and the computational effort tends to become excessive; more direct particle–particle interactions have to be taken into account due to the clustering. This limits the number of particles that can be simulated and the degree of nonlinearity that can be handled.

Another problem due to the grid of the PIC algorithm is that it imposes boundaries. Such boundaries might be rather artificial if the system is in 'free space' or has a complicated geometry. For systems without a real physical boundary, such as the collision of two galaxies, this usually means that one has to construct a grid much larger than the actual space occupied by the particles to avoid unphysical particle-boundary effects.

For strongly coupled systems it would be desirable to use a direct particle–particle force calculation, if the number of simulation particles was not so limited due to the N^2 scaling of the computation time. In the mid-1980s, several workers devised hierarchical schemes to exploit the fact that a particle interacts strongly with its near neighbours, but less detailed information is needed to describe its interaction with more distant particles. The first codes of this kind were developed independently by Appel (1985), Jernighan (1985), and Porter (1985). Although these codes had a nominal $N \log N$ scaling for the computation time, they used neighbour lists and data structures which tended to become tangled, thus introducing errors due to unphysical groupings of particles. This problem was overcome by Barnes and Hut (1986), who used a tree structure rebuilt from scratch at each timestep, ensuring that particle groupings were systematically updated. An appealing feature of the Barnes–Hut scheme is that the $N \log N$ scaling can be rigorously proven.

In Chapter 2, the so-called hierarchical 'tree' algorithm will be described in its basic form. It will be shown how this special means to divide space is used to construct a hierarchical data structure. This tree structure provides a systematic way of determining the degree of 'closeness' between two different particles without explicitly calculating the distance between each particle pair. The net result is to reduce the computational effort of the force calculation to $O(N \log N)$. The force on an individual particle from other particles close by is, on average, evaluated by direct particle–particle interaction, whereas the force due to more distant particles is included as a particle-cluster contribution. To obtain a better accuracy, the multipole moments of the cluster can be included. For dynamical systems, this process of building a tree structure and using it for the force calculation is repeated at each step.

A fairly sizeable literature already exists on tree codes, including a number of works comparing their performance against standard PP codes – notably Hernquist (1987). It is beyond the scope of this book to provide an exhaustive list of 'validity' criteria for hierarchical algorithms, but a few of the more important tests, such as energy and momentum conservation, will be discussed at the end of Chapter 2. Specific examples will be described in Chapters 3 and 6.

In Chapter 3, applications of the tree algorithm will be described in astrophysics, beam transport, and nuclear physics. The examples discussed here have the common feature that the system evolves freely in space and no fixed boundaries are imposed. Open boundary problems are easiest to investigate with tree codes because one simply checks the size of the system after each timestep and adapts the simulation box accordingly so that all particles are included. Tree codes were first developed to study the collision of galaxies (Barnes & Hut 1986, Barnes & Hernquist 1993), where this adaptiveness of the simulation region is of great advantage. Some examples of such simulations which have led to a new understanding of galaxy dynamics and insight into the role of dark matter will be described.

The transport of particle beams in storage rings is of widespread importance for applications requiring a well-characterised and tightly focussed source of electrons or ions. Whether the beam is for etching grooves on an integrated circuit, or for generating X-rays in an inertial fusion hohlraum, it is essential to optimise the quality of the beam as it is accelerated and transported. After a brief introduction to elementary particle beam concepts, some applications are proposed where tree codes could challenge the near-monopoly on beam modelling currently enjoyed by PIC codes.

At one of the frontiers of fundamental physics is the field of heavy ion collisions. By smashing heavy ions together and observing the fragments which result from disintegrating nuclei, one can learn a great deal about how nuclear matter is held together. Through a combination of experiments and theoretical modelling, it is possible to deduce the properties not only of elemental nuclei, but also of more exotic objects and phenomena, such as neutron stars and the early stages of the universe. One promising approach for studying nuclear fragmentation is 'Quantum Molecular Dynamics' (QMD), which explicitly takes into account many-body correlations between nucleons. A detailed case study is presented here, showing how the tree algorithm can be integrated into existing QMD models.

The tree algorithm in Chapter 2 is described in its basic working form. In Chapter 4 it will be shown how to improve the performance of such a code, both through higher accuracy and standard optimisation techniques. A common problem in N-body simulations is that a relatively small number of particles undergo close encounters. Assuming that the necessary stability criteria are satisfied, these particles often determine the timestep on which the whole simulation has to be performed. One way to reduce the computation time employed in direct particle–particle calculations is to use an individual timestep for each particle. Due to the hierarchical structure this trick is not easy to combine with the tree algorithm, though a few implementations now exist. It will be shown that it is possible to have different timesteps for particles undergoing close encounters and particles belonging to the rest of the system, reducing the computation time significantly. Better accuracy of the particle path can be achieved by introducing higher order integration schemes. In addition to these software improvements, the performance of the tree code also depends on the computer hardware. Details will be given on how the tree algorithm can be restructured so that vectorisation and parallelisation architectures can be fully exploited.

There are many cases where it is not possible to simulate an entire physical system (e.g., a solid). On the other hand, it is reasonable to model a small part of it and take the rest into account by including periodic images of the simulation area. These so-called periodic boundary problems are treated in standard particle–particle codes either by the minimum image method (Metropolis et al. 1953) or the Ewald method (Ewald 1921). In the minimum image method, a box is formed around the individual particle, which is equal to the size of the simulation area. In the force calculation only the interactions with particles within this box are included. For the tree algorithm a difficulty arises because the cutting process tends to split the more distant groupings of particles. In Chapter 5 it will be shown how this method nevertheless can be adapted to tree codes. The minimum image method can be used only for weakly coupled systems; for strongly coupled systems the force of more distant particles has to be included too. In this case, the Ewald summation method is required, which includes an infinite number of periodic images by modifying the Coulomb potential – the so-called Ewald potential. Due to the fact that the tree algorithm deals with particle–pseudoparticle as well as particle–particle interactions, it is also necessary to include the higher moments of the multipole expansion of the Ewald potential (Pfalzner & Gibbon 1994).

During the last 15 years a lot of effort has been put into extending the applicability of the MD method to systems out of equilibrium. It will be shown how the tree method can be used for nonequilibrium simulations and where special care is needed to control thermodynamic quantities such as temperature. The properties of the periodic boundary system are often investigated, not only by dynamic methods (or molecular dynamics), but via static or Monte Carlo methods. A brief section is included to show how the tree algorithm can also be used to perform such calculations.

In Chapter 6 examples for the application of periodic tree codes will be described. One obvious candidate for periodic tree simulations is dense plasma. These relatively large systems interact by long-range Coulomb forces. Low density plasmas can be modelled successfully by hydrodynamic codes, which basically treat the plasma as ideal gas, or by PIC codes, which are more suitable for collisionless problems. As the density increases, collisions become important and have to be included in the calculation. Fokker–Planck codes model this by a collision term, but this procedure is usually somewhat artificial, and is valid only for small-angle scattering. Particle–particle codes have the advantage that they include collisions without any artificial assumptions. Static properties such as structure factors have been successfully calculated for very high densities by standard particle–particle codes, but dynamical properties and strongly nonlinear effects need a much larger number of simulation particles. It is precisely this sort of application for which periodic tree codes would make an ideal simulation tool.

In addition to these systems which interact purely by Coulomb forces the tree algorithm can be applied successfully to systems interacting via different force terms like in Eq. 1.2 where one of the terms is either gravitational or Coulombic. In these cases, the long-range term is usually by far the most time consuming and a speed-up by means of the tree algorithm can bring large gains. Different applications of tree codes to systems with a more complex structure of the potential are proposed here, among them ionic liquids, molten salts, and biological macromolecules like proteins.

Chapter 7 contains an introduction to the so-called Fast Multipole Method (FMM), which is, in some sense, based on a hierarchical tree code using a high-order multipole expansion. This kind of code is mathematically more complicated and has a higher computational effort than a standard tree code, but the computation time has in principle an $O(N)$ dependence. Formulations of the algorithm for both 2D and 3D problems will be outlined, based on works by Greengard and Rohklin (1987), and Schmidt and Lee (1991). An attempt is also made to compare the relative performance of the BH and FMM algorithms.

2

Basic Principles of the Hierarchical Tree Method

2.1 Tree Construction

We have seen in the preceding chapter that in grid-based codes the particles interact via some averaged density distribution. This enables one to calculate the influence of a number of particles represented by a cell on its neighbouring cells. Problems occur if the density contrast in the simulation becomes very large or the geometry of the problem is very complex.

So why does one bother with a grid at all and not just calculate the inter-particle forces? The answer is simply that the computational effort involved quite dramatically limits the number of particles that can be simulated. Particularly with $1/r$-type potentials, calculating each particle–particle interaction requires an unnecessary amount of work because the individual contributions of distant particles is small. On the other hand, gridless codes cannot distinguish between near-neighbours and more distant particles; each particle is given the same weighting.

Ideally, the calculation would be performed without a grid in the usual sense, but with some division of the physical space that maintains a relationship between each particle and its neighbours. The force could then be calculated by direct integration while combining increasingly large groups of particles at larger distances. Barnes and Hut (1986) observed that this works in the same way that humans interact with neighbouring individuals, more distant villages, and larger states and countries. A resident of Lower-Wobbleton, Kent, England, is unlikely to undertake a trip to Oberfriedrichsheim, Bavaria, Germany, for a beer and to catch up on the local gossip.

Independently, in the early 1980s, several workers attempted to implement this kind of hierarchical grouping in N-body codes (Appel 1985, Jernighan 1985, Porter 1985). Although these early hierarchical codes had a nominal $N \log N$ dependence of the computation time, additional errors were introduced