Asci'96 35

Parallel Hierarchical Methods for Large Scale Simulations of Virtual Particle Models

A.G. Hoekstra and P.M.A. Sloot
Parallel Scientific Computing & Simulation group
Faculty of Mathematics, Computer Science, Physics, and Astrophysics
University of Amsterdam
email: {alfons, peterslo}@fwi.uva.nl
url: http://www.fwi.uva.nl/research/pwrs/

Keywords. Hierarchical Tree Methods, Barnes and Hut, Fast Multipole Method, N-body methods, Molecular Dynamics, Computational Electromagnetics, Radiosity.

Introduction

An important research item in our group is the development of generic computational techniques, which allow efficient simulations of a wide range of systems. Sloot et al. introduced the concept of Dynamic Complex Systems (DCS) as an abstract and generic approach to parallel computing in large scale simulations [1]. The DCS approach allows direct simulations of physical phenoma, through the use of so-called natural solvers. Subsequently it allows the identification of locality in the solvers, which is a necessary requirement for efficient parallel computing.

An important generic computational model is characterized by a set of mutually interacting entities, which we coin Virtual Particles. Virtual Particle Models can simulate a wide range of systems, from the N-Body system to Radiosity in computer graphics. At first sight Virtual Particle Models are straightforward to implement and to parallelize. However, for large scale simulations, containing many virtual particles, the straightforward approach is seriously hampered by its quadratic time complexity in the number of particles.

This paper describes our ongoing work in the field of hierarchical methods, which we use to reduce the computational complexity of the Virtual Particle Model, and efforts to parallelize the hierarchical algorithms, resulting in the possibility of large scale simulations (i.e. containing a very large number of particles) of the Virtual Particle Model. First, we will shortly describe three application domains where we exploit the Virtual Model abstraction for our simulations. Next, we introduce hierarchical methods, and finally we will mention our ongoing work in parallilization of hierarchical methods.

The Virtual Particle Model

The Virtual Particle Models is characterized by a set of mutually interacting particles. Traditionally, the N-body system with long range interaction is the best known and most widely studied member of this class of applications. The entities can be massive bodies (stars or galaxies) which interact through the gravitational force, or can e.g. be ions in a polar fluid which interact through the Coulombic electrostatic force. Simulations in terms of such N-body systems are collectively known as particle methods [2]. In particle methods the dynamic behaviour of a system is simulated, and relevant macroscopic quantities (e.g. global structure of a galaxy, thermodynamic properties of a fluid) are extracted from the trajectories of the constituent particles. Each iteration of a particle simulation consists of the calculation of the force on each particle, due to all other particles, and an update of the particle positions from time t to time $t + \Delta t$.

Outside the domain of particle methods many important applications exist which share the same characteristics. For instance, a widely used method in Computational Electromagnetics to solve Maxwell's equations can be viewed as a system consisting of interacting radiating dipoles. In this case the system is not advanced in time during each pass of the simulation, but is advanced over an abstract iteration-axis towards a situation of a self-consistent electromagnetic field on each dipole [3,4]. The radiosity method in computer graphics is comparable. Here, a self-consistent radiosity on all polygons which build up a scene must be obtained.

Assume that we wish to simulate a system containing N virtual particles. For each pass of the simulation loop the complete interaction between all entities must be obtained (e.g. the gravitational force on each star due to all other stars). Obviously this operation has an $O(N^2)$ complexity, and is the most time consuming operation in this type of simulations. An efficient parallel implementation of this interaction calculation is straightforward (see e.g. reference [5], chapter 9). However, the $O(N^2)$ complexity is prohibitive if large systems (e.g. $N > 10^4$) have to be studied. Therefore, many attempts have been made to reduce this complexity. A very successful class of methods are the hierarchical tree methods, which reduce the complexity to $O(N\mathrm{Log}N)$ or even to $O(N)$, and which have been developed in the realm of the particle methods.

Hierachical Methods

Hierarchical Methods were developed for particle simulations, which all share a common $1/r$ interaction potential, where r is the distance between two particles. The general idea is to group particles together and describe their total interaction with a particle as a single multipole series. By doing this in a hierarchical way, such that groups of remote particles are contained in increasingly larger subvolumes of the computational box, the complexity of the interaction calculation can be reduced drastically.

Currently two formulations are known. The first is the Barnes and Hut method [6], which results in an $O(N\mathrm{Log}N)$ complexity, and the Fast Multipole Method of Greengard [7], which has $O(N)$ complexity. In both methods the computational box containing the particles is recursively subdivided into smaller and smaller subcubes, until, at the finest level a subbox contains a small number of particles (usually 1). The resulting datastructure is an octree. All hierarchical methods contain three routines which act on the octree datastructure:

- building of the octree,
- bottum up pass, where the multipole moments in the subcubes are calculated,
- top down pass, where interaction between particles and the largest possible

subcubes is calculated.

We have defined a template for hierarchical algorithms, which can express both the Barnes and Hut method (BH) and the Fast Multipole Method (FMM). Furthermore, we can also easily express hybrids of the BH and FMM method, which can be of importance for several types of applications. We have implemented and tested a sequential version of this template on the PowerXplorer, and we are currently in the process of developing an efficient parallel implementation of hierarchical methods.

We have developed hierarchical methods for other application areas, such as Computational Electromagnetics [4]. Furthermore, hierarchical methods for Radiosity have been reported in the literature [8]. We will show examples of these algorithms, and show how they can be incoorperated in our template for hierarchical methods.

Parallelizing Hierarchical Methods

Development of efficient parallel implementations for hierarchical methods is an active research area. Several examples of parallel implementations are described in the literature. However, we observe that no generally accepted method to build parallel hierarchical methods exists, especially if the target architecture is a distributed memory system. Therefore, we are currently implementing a parallel version of the template for hierarchical methods on the PowerXplorer system in Amsterdam. The hard- and software of the PowerXplorer system are very well suited to adress the many open research questions in parallel hierarchical methods.

Basically, the design choices for a parallel hierarchical method are twofold. First, it must be decided how the calculation is decomposed. The decomposition has to guarentee load balancing, and on distributed memory systems also data locality. Here we have two options. One can apply a geometric decomposition of the computational box, using e.g. Orthogonal Recursive Bisection (ORB) or Recursive Spectral Bisection (RSB). The basic problem now is to embed the octree datastructure into the resulting geometric decomposition. One can also try to perform a data decompostion on the octree itself. Several approaches are known in the literature. First, one can take a top-down approach, and give complete subtrees to processors. This approach was taken by e.g. Leathrum et al. in parallel versions of the FMM method [9]. Here, loadbalancing is not guarenteed, especially in non-uniform particle distributions. Other methods take a bottom-up approach, where the leafs of the octree, containing the particles, are distributed among the processors. In this case loadbalancing is much easier to achieve, but now the problem of data locality becomes an important issue. Two techniques are known, the costzones of Pal Singh [10], and the Hashed Octree Table (HOT), using Morton ordening, of Salmon and Warren [11]. In all cases it is far from trivial to perform the decomposition in parallel. Furthermore, as the octree continuously changes due to the dynamics of the underlying particle system, regular updates of the datastructure are necessary, giving rise to very complex issues such as dynamic loadbalancing.

Next, if a suitable decomposition has been achieved, another important design choice has to be made, which in one case results in a loosely synchronous implementation, and in the other case in a completely asynchronous parallel program. In the first approach, the structure of the sequential algorithm is maintained and first the bottomup phase, where the multipole moments in the subcubes are calculated, is performed. During this phase each processor will send data which *might* be needed by other processors in the next phase of the program to the appropriate processors. At this point each processor has all essential data to proceed with the next step, which is the topdown phase, where the force on each particle in a processor is calculated. This step can be done completely in parallel, since all processors have all necessary information available. An example of this approach is the Locally Essential Tree (LET) method of Salmon [12].

The other approach combines the bottum-up and top-down phase. Each processor starts calculating, and if it needs data from other processors, this data is requested. In the mean time the processor continues with other parts of the calculation and after a certain period it cheques if the requested data has been received. If this is the case, the postponed calculation is finished. This approach results in a completely asynchronous, demand-driven parallel program. Currently, just one example of this approach is described in the literature [11]. Is is claimed that, due to the possibility of latency hidding, this approach results in very efficient parallel hierarchical methods, especially for very large systems.

We will report on our parallel version of the BH hierarchical method on the PowerXplorer, using the ORB decomposition of the computational box, and the LET technique of Salmon. Currently, we are evaluating this implementation in the context of Molecular Dynamics simulations of nucleation processes in fluids.

Our current implementation uses a geometric decomposition of the computational box, and the LET technique, resulting in a loosely synchronous program. We will also implement the HOT decomposition in combination with the demand driven approach.

References

1] P.M.A. Sloot, J.A. Kaandorp, and A. Schoneveld, "Dynamic Complex Systems (DCS): a new approach to parallel computing in computational physics" Technical Report CS-95-08, department of Computer Science, University of Amsterdam, 1995.

2] R.W. Hockney and J.W. Eastwood, *Computer Simulation Using Particles*, IOP Publishing, 1988 (Bristol, New York).

3] A.G. Hoekstra, *Computer Simulation of Elastic Light Scattering, implementation and applications*, Ph.D. thesis, University of Amsterdam, the Netherlands, 1994.

4] A.G. Hoekstra and P.M.A. Sloot, "Computational Electromagnetics using Virtual Particles", International Journal in Computer Simulation, special issue on "New Techniques of Computer Simulation in the Natural Sciences", 1996, in press.

5] G. Fox et al., *Solving Problems on Concurrent Processors volume I, general techniques and regular problems*, Prentice Hall International Editions, 1988.

6] J. Barnes and P. Hut, "A hierarchical O(NlogN) force-calculation algorithm", nature vol. 324, 446, 1986.

7] L.L. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems* , the MIT Press, 1988.

8] P. Hanrahan, D. Salzman, and L. Aupperle, "A Rapid Hierarchical Radiosity Algorithm", in Proceedings of SIGGRAPH, 1991.

9] J.F. Leathrum, *Parallelization of the Fast Multipole Algorithm: algorithm and architecture design*, Ph.D. thesis Duke University, USA, 1992.

10] J.P. Singh, *Parallel Hierarchical N-Body Methods and their Implications for Multiprocessors*, Ph.D. thesis Stanford University, USA, 1993.

11] M.S. Warren and J.K. Salmon, "A Portable Parallel Particle Program", Computer Physics Communications, vol. 87, pp. 266-290, 1995.

12] J.K. Salmon, *Parallel Hierarchical N-Body Methods*, Ph.D. thesis, California Institute of Technology, USA, 1991.