Astronomical Data Analysis Software and Systems X
ASP Conference Series, Vol. 238, 2001
F. R. Harnden Jr., F. A. Primini, and H. E. Payne, eds.

An Integrated Procedure for Tree N-body Simulations: FLY and AstroMD

U. Becciani, V. Antonuccio-Delogu

Osservatorio Astrofisico di Catania, Catania - Italy

F. Buonomo, C. Gheller

Cineca, Casalecchio di Reno (BO) - Italy

Abstract. We present a new code for evolving three-dimensional self-gravitating collisionless systems with a large number of particles $N \geq 10^7$. **FLY** (Fast Level-based N-bodY code) is a fully parallel code based on a tree algorithm. It adopts periodic boundary conditions implemented by means of the Ewald summation technique. FLY is based on the one-side communication paradigm for sharing data among the processors that access remote private data, avoiding any kind of synchronization. The code was originally developed on a CRAY T3E system using the *SHMEM* library and it was ported to SGI ORIGIN 2000 and IBM SP (on the latter making use of the *LAPI* library). FLY[1] version 1.1 is open source, *freely available code*.

FLY output data can be analysed with AstroMD, an analysis and visualization tool specifically designed for astrophysical data. AstroMD can manage different physical quantities. It can find structures without well defined shape or symmetries, and perform quantitative calculations on selected regions. AstroMD[2] is *freely available*.

1. Introduction

FLY is the N-body tree code we designed and developed to run very big Large Scale Structure simulations of the universe using MPP and SMP parallel systems. FLY uses the Leapfrog numerical integration scheme for performance reasons, and incorporates fully periodic boundary conditions using the Ewald method. The I/O data format is integrated with the AstroMD package.

AstroMD is an analysis and visualization tool specifically designed for astrophysical data. AstroMD can find structures not having a well defined shape or symmetries, and perform quantitative calculations on a selected region or structure. AstroMD makes use of Virtual Reality techniques, which are particularly effective for understanding the three dimensional distribution of the

[1] `http://www.ct.astro.it/fly/`

[2] `http://www.cineca.it/astromd`

fields: their geometry, topology, and specific patterns. The data display gives the illusion of a surrounding medium into which the user is immersed. The result is that the user has the impression of traveling through a computer-based multi-dimensional model which can be manipulated by hand.

2. FLY Code

The FLY code, written in Fortran 90 and C languages, uses the one-side communication paradigm: it has been developed on the CRAY T3E using the SHMEM library. It adopts a simple domain decomposition, a grouping strategy, and a data buffering that allows us to minimize data communication.

2.1. Domain Decomposition

FLY does not split the domain with orthogonal planes. Instead, domain decomposition is done by assigning an equal number of particles to each processor. The input data are a sorted file containing the position and velocity fields, so that particles with nearby tag number are also close in the physical space. The arrays containing the tree properties are distributed using a fine grain data distribution.

2.2. Grouping

During the tree walk procedure, FLY builds a single interaction list (IL) to be applied to all particles inside a grouping cell (C_{group}). This reduces the number of tree accesses required to build the IL. We consider a hypothetical particle we call *Virtual Body* (VB) placed in the center of mass of the C_{group}: the VB interaction list IL_{VB} is formed by two parts:

$$IL_{VB} = IL_{far} + IL_{near} \tag{1}$$

where IL_{far} includes the elements more distant than a threshold parameter from VB, and IL_{near} includes the elements near VB. Using the two lists, it is possible to compute the force F_p of each particle p in C_{group} as the sum of two components:

$$F_p = F_{far} + F_{near} \tag{2}$$

The component F_{far} is computed only once for VB, and it is applied to all the particles, while the F_{near} component is computed separately for each particle. The size of the C_{group}, and the tree-level where it can be considered, is constrained by the maximum allowed value of the overall error of this method. In this sense the performance of FLY is a level-based code.

2.3. Data Buffering

The data buffer is managed as a simulated cache in local RAM. Every time the PE has to access a remote element, at first it looks in the local simulated cache and, if the element is not found, the PE executes GET calls to download the remote element, and stores it in the buffer. In a simulation with 16-million-particles clustered, with 32 PEs and 256 MB of local memory, without the use of the simulated cache, the PEs execute about 2.1×10^{10} remote GETs. Using the

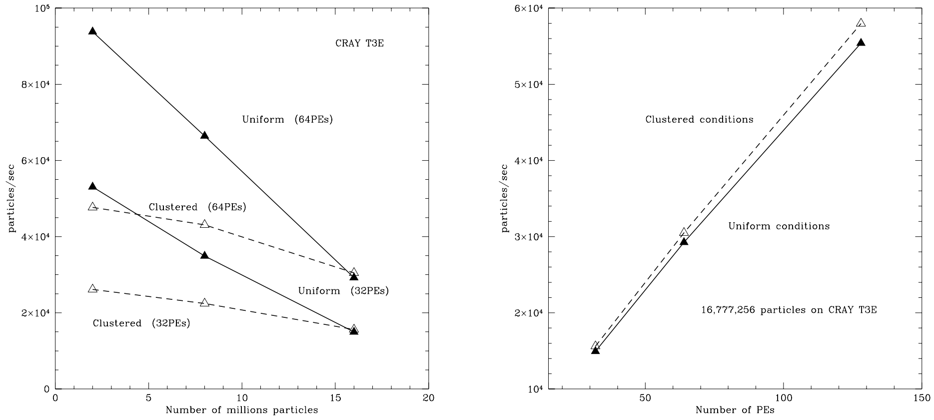

Figure 1. CRAY T3E/1200e. FLY particles/second using 32 and 64 PEs in uniform and clustered conditions, and scalability from 32 PEs to 128 PEs in a 16-million-particle simulation.

data buffering, this value decreases at 1.6×10^8 remote GETs, with an enormous advantage in terms of scalability and performance.

Figure 1 shows the code performance on CRAY T3E system, obtained by running simulations with 32 and 64 PEs, and FLY scalability, considering the case of 16,777,216 particles, where a speed-up factor of 118 is reached using 128 PEs. The highest performance obtained in a clustered configuration is a positive effect of the grouping characteristic. These results show that FLY has very good scalability and very high performance, and can be used to run very big simulations.

3. AstroMD Package

AstroMD is developed using the Visualization Toolkit (VTK) by Kitware, a freely available software portable to several platforms which range from the PC to the most powerful visualization systems, with a good scalability. Data are visualized with respect to a box which can describe the whole computational mesh or a sub-mesh. AstroMD can find structures not having well defined shape or symmetries, and performs quantitative calculations on a selected region or structure. The user can choose the sample by the loaded particle type (i.e., stars, dark matter, and gas particles), the size of the visualization box, and the starting time from which he wants to show the evolution of the simulation. The *Density* entries control the visualization of the iso-surfaces. These are calculated on a grid whose resolution can be selected by the user. To allow a more accurate investigation of a subset of the visualized system, it is possible to use a cubic sampler. If the sampler is selected (*Show/Hide* menu), visualized in the scene and enabled (*Sampler* menu), all the computations are performed only inside the region of the sampler (Figure 2). AstroMD can also show the evolution in time of the simulated system over the entire interval of time for which data are available, performing interpolations at intermediate frames. During the evolu-

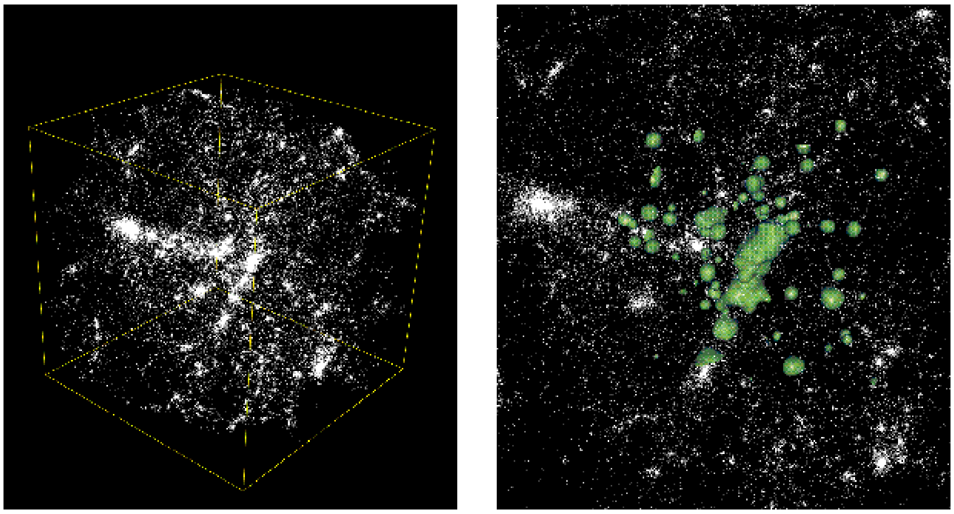

Figure 2. Typical structure of voids and filaments and Iso-surfaces, in a sub-region at the end of a simulation.

tion, the updated time of evolution is displayed in the *Time* entry of the *Cloud* section. Finally, snapshots of displayed images can be created using the button *Take it* in the *Screenshot* menu. AstroMD is developed by the VISIT (Visual Information Technology) laboratory at CINECA (Casalecchio di Reno - Bologna) in collaboration with the Astrophysical Observatory of Catania.