Bull. Nov. Comp. Center, Comp. Science, 19 (2003), 73–81

A parallel program for simulation of disc-shaped self-gravitating systems*

A.V. Snytnikov

A model of self-gravitating system is described. Present parallel programs implementation of such a model are reviewed in brief. Numerical algorithm is described. The parallelization technique and load balancing strategy are discussed in detail. The parameters of test computations that meet the requirements of the problem of protoplanetary disc simulation are given.

1. Introduction

The evolution of self-gravitating systems such as galaxies and protoplanetary discs is of great interest to astrophysics: about 20% of the papers are devoted to this problem, as referred in [1]. The problem is essentially the N-body problem in self-consistent gravitational field. A good approximation for this problem is Vlasov–Liouville kinetic equation. Together with the Poisson equation they are called the stellar dynamics equations [2]:

$$\Delta\Phi = 4\pi G\Sigma\delta(z), \quad \frac{\partial f}{\partial t} + \vec{v}\nabla f - \nabla\Phi\frac{\partial f}{\partial \vec{v}} = 0,$$

where Φ is the gravitational potential, Σ is the surface density, $\delta(z)$ is the Dirak function, f is the distribution function or phase density, $\vec{v}$ is particles' velocity. Here the disc is considered flat and collisionless.

An accurate solution of this system of equations requires the application of supercomputers. It could be clarified by the following estimate. Let us consider, for example, chemical evolution in the Solar System. The most interesting processes take place inside Mars orbit, which is fifty times smaller than the radius of the whole system. If we take ten nodes by the radial coordinate for the length of Mars orbit radius, we obtain then five hundred nodes of the radial grid. Because the boundary conditions for the Poisson equation are to be set far from the actual boundary of the simulation domain, the number of radial nodes is to be doubled. Furthermore, the grid cell should be approximately cubical. It means that the number of nodes by

*Supported by SB RAS integration project under Grant 43, INCO-COPERNICUS program under Grant 97-7120, and Russian Foundation for Basic Research under Grants 99-07-90422 and 02-01-00864.

second planar coordinate should be of the same value and by vertical coordinate about one half of that. Thus we have an estimate of spatial grid size for the present problem as 5×10^8 nodes. If each node is represented by one double precision number, then 4 Gb RAM are needed for a grid of this size. Let us note that when the grid is increased to N times, the computation time grows as N^γ, where $\gamma > 1$.

The present work has two goals. First, the design of a program which provides high resolution at a reasonable worktime. Second, the achievement of good speedup and resource-to-worktime ratio. It means that at first we want to perform our simulations with high precision. Moreover, the program should work as fast as it is possible at the given amount of resources.

2. Gravitational solvers review

There are several techniques for numerical solution of stellar dynamics equations. The first is the direct evaluation of the particles' interaction force, the so called "particle–particle" method or P^2. This method is the most precise, but also the most time-consuming, its complexity is $O(N^2)$, where N is the number of particles. Evaluation of a particle's movement by this method requires positions of all the rest particles. Thus a simulation employing more than 10^5 particles is impossible even with a supercomputer.

A variant of P^2 method is the treecode, the method in which closely situated particles are united in groups in order to treat them as one particle of corresponding mass in the gravitational force evaluation. If such force approximation is inaccurate, the group is divided into smaller subgroups. Decomposition of particles into groups has the shape of an octal tree, which gives the name to the method. This method is significantly faster than P^2, its complexity being $O(N \log N)$, but less accurate. Distribution of the processor workload is done during the tree setting [6].

According to the "particle–mesh" method, or the PM a grid is introduced in the simulation domain. Density is computed on the grid by interpolation of the particles' masses from the positions of the particles. With the given density gravitational potential is computed by the Poisson equation solution: generally the FFT method is employed. The force acting at a particle is evaluated by interpolation of the grid values into the particle's position. The PM method is the fastest one, its complexity is only $O(N \log M)$, M being the number of grid nodes. The disadvantage of this method is that it gives adequate accuracy only for uniform systems – otherwise an extremely large number of particles is required. For parallelization of the method the main difficulty is in the Poisson equation solver. If the main time is consumed by computation of the particles' movement it is necessary to use dynamic load balancing [5]. The algorithm for dynamic load balancing proposed in [7] is

based on the problem decomposition into a set of smaller problems which are assigned to unloaded processors.

There is a number of programs for cosmological simulation implementing P^3M method. This method is a combination of P^2 and the PM methods: the interaction between closely situated particles is evaluated directly, whereas the force acting from distant particles is computed via the Poisson equation. The application of P^3M method to cosmological problems could be explained by the following reason: in this case the matter is divided into isolated dense subdomains. During parallelization of the P^3M method in [8] two different schemes of decomposition are involved: one to compute the interaction of particles (P^2 part) and the other to solve the Poisson equation (the PM part). The complexity of this method varies from $O(N \log M)$, when density distribution is uniform, to $O(N^2)$, when the matter is strongly clumped and evaluation of particles' couple interaction takes the main time.

A combination of all mentioned methods is the method called the TPM ("Treecode–Particle–Mesh") [9]. This algorithm is based on the fact that the density field could be broken into isolated dense subdomains. In each subdomain gravitational potential is computed via treecode, while force and potential given by the rest of the domain are evaluated by the PM method. Since every subdomain could be treated independently the algorithm is very well parallelized. The trees corresponding subdomains are distributed between the processors to make their workload equal. Comparison showed that the TPM algorithm works with at least the same accuracy as P^3M, but much faster. The efficiency of parallelization depends on clumping of the matter, that is, on the number and size of the trees. Like the previous case, the complexity of this method varies from $O(N \log M)$ to $O(N \log N)$.

Table 1

Method	Grid size	Number of processors	ξ, %	Number of particles	Paper
P^2	–	112	–	3×10^3	[10]
PM	256^3	64	85	16.7×10^6	[7]
Treecode	–	16	93	10^5	[6]
P^3M	1024^3	500	–	10^9	[8]
TPM	512^3	128	90	1.34×10^8	[9]

A comparison of implementation parameters for above mentioned methods is given in Table 1. Here the parallelization efficiency is the ratio of computation speedup to increase of the number of processors. Let T_1 be the worktime on N_1 processors and T_2 the worktime on N_2 processors. Then parallelization efficiency ξ is evaluated this way

$$\xi = \frac{T_1}{T_2}\frac{N_1}{N_2} \times 100\%.$$

3. Numerical algorithm description

The main difficulty in the problem of protoplanetary disc evolution simulation is the computation of the self-consistent gravitational field potential, which is determined by the Poisson equation. In order to obtain the fast Poisson equation solver the peculiarities of the problem are to be considered and the algorithm is to be highly parallelizable. That means it must be able to employ any desired number of processors.

Let us consider the peculiarities of the problem under study. First, it is non-stationary. It means that the physical quantities like gravitational potential, density and velocity field cannot alter drastically. The thing is the values of potential at neighboring time steps cannot be significantly different. Therefore a Poisson equation solver is necessary that can take into account values obtained at previous timestep. The methods which meet this requirement are the iterational ones. Since they are too slow a combined solver is offered which incorporates both direct and iterational methods.

Second, a protoplanetary disc may be considered flat, its thickness being far less than radius and density being nonzero only at the disc surface. It means that the alteration of density due to particles' shift at a timestep leads to a significant potential alteration only at the disc surface. Unlike that at the points which are remote from the disc these alterations are subtle and convergence is reached much faster. Thus computation time is reduced greatly because potential at the points remote from disc surface should not be recomputed at each timestep.

The Poisson equation is solved on a grid in cylindrical coordinate system in order to take disc symmetry into account and rule out the non-physical structures appearing in the Cartesian coordinates. It has the following form

$$\begin{aligned}
&\frac{1}{h_r^2 r_{i-1/2}}\Big[r_i(\Phi_{i+1/2,k-1/2,l}-\Phi_{i-1/2,k-1/2,l})- \\
&\qquad r_{i-1}(\Phi_{i-1/2,k-1/2,l}-\Phi_{i-3/2,k-1/2,l})\Big]+ \\
&\frac{1}{h_\varphi^2 r_{i-1/2}^2}\Big(\Phi_{i-1/2,k+1/2,l}-2\Phi_{i-1/2,k-1/2,l}+\Phi_{i-1/2,k-3/2,l}\Big)+ \\
&\frac{1}{h_z^2}\Big(\Phi_{i-1/2,k-1/2,l+1}-2\Phi_{i-1/2,k-1/2,l}+\Phi_{i-1/2,k-1/2,l-1}\Big)=0, \\
&\quad i=1,\ldots,I_{\max},\quad k=1,\ldots,K_{\max},\quad l=1,\ldots,L_{\max}-1.
\end{aligned}$$

where $I_{\max}$, $K_{\max}$, $L_{\max}$ are the numbers of nodes along radial, angular and vertical coordinates correspondingly; i, k, and l are the numbers of the node along these coordinates; $r_{i-1/2}$ is the radial coordinate of the i-th node; h_r, h_φ, and h_z are the grid steps.

Figure 1 shows the general scheme of the method. The first stage is the Fast Fourier Transform along the angular coordinate resulting in a system of linear algebraic equations. Each of the equations describes only one harmonic of the potential:

$$\frac{1}{h_r^2 r_{i-1/2}}\Big[r_{i-1}H_{i-3/2,l-1/2}(m) + r_i H_{i+1/2,l-1/2}(m)\Big] +$$
$$\frac{1}{h_z^2}\Big[H_{i-1/2,l-3/2}(m) + H_{i-1/2,l+1/2}(m)\Big] -$$
$$\frac{2}{h_\varphi^2 r_{i-1/2}^2}\left[1 + 2\sin^2\frac{\pi m}{K_{\max}}\right]H_{i-1/2,l-1/2}(m)$$
$$= 4\pi R_{i-1/2,l-1/2}(m)\cos\frac{2\pi k m}{K_{\max}},$$
$$m = 1, \ldots, K_{\max},$$

where m is the number of harmonic or angular wavenumber, all the other symbols have the same meaning as in the previous equation.

These equations are completely independent from each other, which is the most important fact.

During the second stage the two-dimensional equations are solved via Successive Over-Relaxation method. Along the radial coordinate the sweeping procedure is applied to decrease the number of iterations. When convergence is reached the inverse Fourier Transform is done on potential harmonics at the disc surface.

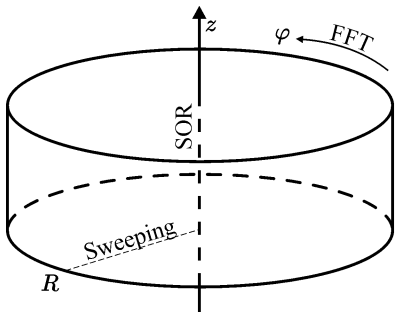

Figure 1. Structure of the Poisson equation solver

The Vlasov–Liouville kinetic equation is solved by Particle-In-Cell (the PIC) method. The scheme of the method is considered in detail in [4].

4. Parallelization

One of the challenges of parallelization is to minimize data exchange between processors. The considered Poisson equation solver succeeds to avoid data exchange completely. That is because equations for potential harmonics do not depend on each other. Therefore it is possible to divide the computation domain into completely independent subdomains along angular wavenumbers as shown in Figure 2. Domain decomposition is uniform, each processor gains equal number of harmonics.

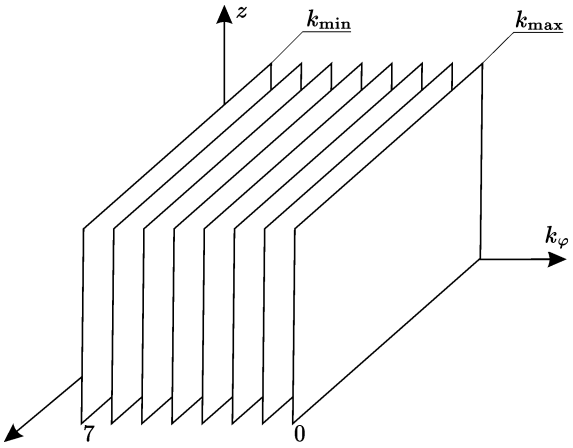

Figure 2. Domain decomposition

Particles are also uniformly between processors with no dependency on their spatial location. Since a particle could fly to any place of the disc in course of simulation every processor should know the potential at all the disc surface.

Interprocessor communications in the program were implemented via collective functions of the MPI library. At each timestep data exchange is performed twice. First, after reaching convergency potential harmonics are gathered for inverse Fourier transform. Then the parts of density field computed at each processor are added up.

5. Test computations

Debug of the program was performed at a Linux workstation with two Pentium III processors. All the test computations were conducted on MVS-1000M supercomputer based on Alpha 21264 processor at Siberian Supercomputer Centre, Novosibirsk and at Joint Supercomputer Centre, Moscow. Parameters of some computations are given in Table 2.

The plots in Figure 3 show the speedup for the test grid having $400\times512\times200$ nodes with 2×10^7 particles. Thus for small number of processors the

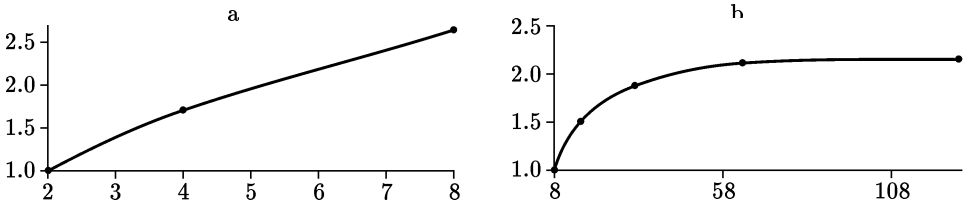

Figure 3. Speedup on the test grid

Table 2

Number of processors	Grid size			Number of particles	Worktime for one timestep, s
	N_R	N_φ	N_Z		
2	400	512	200	2×10^7	19.0
4	400	512	200	2×10^7	11.1
8	400	512	200	2×10^7	7.2
8	400	512	200	10^8	28.0
64	1000	1024	1000	10^9	141.0
128	500	1024	400	10^8	204.0
64	1000	2048	800	4×10^8	229.8
128	1000	2048	800	4×10^8	180.0
256	1000	2048	800	4×10^8	177.6

efficiency of parallelization was 85% (see Figure 3a). The speedup becomes smaller as number of processors increases (see Figure 3b) and for more than 128 processors almost vanishes. Such a phenomenon is easily explained because the workload of the processor is not equal in fact. The assignment of harmonics to processors is uniform, but different harmonics require different number of iterations as it is illustrated by the plot in Figure 4.

The 10th timestep is shown, when the disc is nearly axysymmetric: few long harmonics dominate in computation time. Density distribution initially has the axial symmetry. Therefore after the FFT only 0th harmonic is not equal to zero. Convergence for all the rest harmonics is reached in one iteration. It is clear that on the first timestep only one processor is working. In the course of simulation the symmetry is being lost and it requires time to compute all the harmonics. Still the long harmonics require more time than short ones.

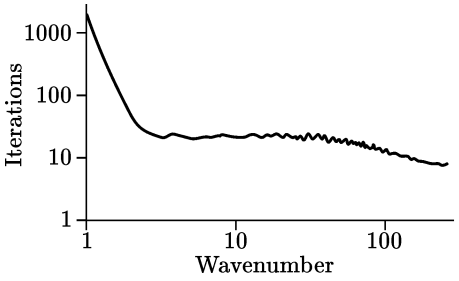

Figure 4

It should be noted that speedup is different for different grids. On the plot above the difference in worktime between 64 and 128 is only 2%. When the grid gets larger the computation time grows faster than communication time. The same difference for a larger grid with $1000 \times 2048 \times 800$ nodes (see Table 2) is 10%.

6. Dynamic load balancing

The problem of load imbalance due to different condition numbers of different harmonics could be solved in the following way. Minimal worktime

is achieved when the harmonic requiring maximal number of iterations is the only workload a processors. Let us call the workload of this processor maximal. Furthermore, the workload of all the rest processors should not exceed the maximal one.

Thus the load balancing procedure could be obtained this way. After finishing all the iterations in each processor the most loaded one is found. Then the harmonics are transferred from it to the least loaded processor while such a relationship of workloads remains. That is, when the receiver is no more the least loaded, a new receiver is found if the sending processor is still overloaded.

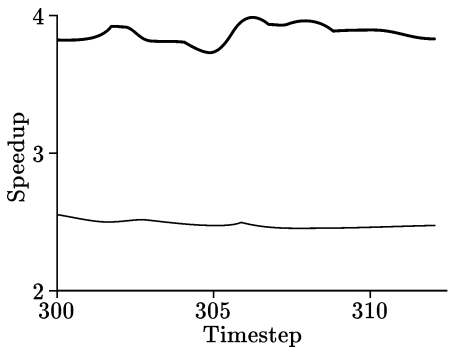

Figure 5. Comparison of speedup for static load setting (thin line) and for dynamic load balancing (thick line)

The disadvantage of this approach is clear: balancing is made for the past step, because the number of iterations becomes known only when the computations are over. However, one can rely on that the number of iterations for each harmonic will not differ greatly at adjacent timesteps. It means that load balancing could be made in the above described manner.

The plot in Figure 5 shows the speedup for several adjacent timesteps of a simulation (from 300th to 312th step) with dynamic load balancing and with static load setting.

7. Conclusion

Conducted were the computations with parameters meeting the requirements of the problem of protoplanetary disc simulation. These computations make it possible to state that the designed program is competitive with world analogues. Parallelization efficiency is 75% up to 32 processors which is due to non-uniform processor workload. Whereas the dynamic load balancing increases the speedup close to the ideal value, and thus greatly improves efficiency.

Acknowledgements. I would like to thank my supervisors V.E. Malyshkin and, especially, V.A. Vshivkov.

References

[1] Ellis R. Galaxy formation and evolution: recent progress // Lectures in the XIth Canary Islands Winter School on Astrophysics, 2001.

[2] Bertin G. Dynamics of Galaxies. – Cambridge University Press, 2000.

[3] Hockney R.W., Eastwood J.W. Computer Simulation Using Particles. – Bristol: IOP Publishing, 1988.

[4] Snytnikov V.N., Vshivkov V.A., Dudnikova G.I., Nikitin S.A., Parmon V.N., Snytnikov A.V. Numerical simulation of N-body gravitational systems with gas // Vychislitel'nye Tehnologii. – 2002. – Vol. 7, № 3. – P. 72–84.

[5] Kraeva M.A., Malyshkin V.E. Implementation of PIC method on MIMD multicomputers with assembly technology // Proc. of HPCN Europe 1997, LNCS. – Springer Verlag, 1997. – Vol. 1255. – P. 541–549.

[6] Miocchi P., Capuzzo-Dolcetta R. An efficient parallel tree-code for the simulation of self-gravitating systems // A&A. – 2001. – http://xxx.lanl.gov/astro-ph/0104152.

[7] Caretti E., Messina A. Dynamic work distribution for PM algorithm. – 2000. – http://xxx.lanl.gov/astro-ph/0005512.

[8] MacFarland T., Couchman H.M.P., Pearce R.F., Pilchmeier J. A new parallel P3M code for very large-scale cosmological simulations // New Astronomy. – 1998. – http://xxx.lanl.gov/astro-ph/9805096.

[9] Bode P., Ostriker J., Xu G. The tree-particle-mesh N-body gravity solver // ApJS. – 1999. – http://xxx.lanl.gov/astro-ph/9912541.

[10] Griv E., Gedalin M., Liverts E., Eichler D., Kimhi Ye., Chi Yuan Particle modeling of disk-shaped galaxies of stars on nowadays concurrent supercomputers // NATO ASI "The Restless Universe". – 2000. – http://xxx.lanl.gov/astro-ph/0011445.