Mem. S.A.It. Suppl. Vol. 4, 46
© SAIt 2004

Memorie della
Supplementi

Parallelization of a Smoothed Particle Hydrodynamic Code for Simulation of Shocks in Accretion Disks

G. Gerardi and D. Molteni

Dipartimento di Fisica e Tecnologie Relative, Università di Palermo, Viale delle Scienze (ed.18), Palermo e-mail: `gerardi@math.unipa.it, molteni@unipa.it`

Abstract. We present the parallelization of a Serial Code that was designed to model shocks around black hole in accretion disks using the Smoothed Particles Hydrodynamics (SPH). The goal we obtained with our parallelization is threefold: (1) Execution speed gain nearly proportional to the number of processors; (2) Processing a number of SPH particles so high that doesn't fit into a single computer; (3) Implementation of the parallel code in such a way to make future additions of new physic ingredients easier. The paradigm we use to realize the parallelization is the Multiple Instruction Multiple Data (MIMD) one. The three dimensional computational domain is decomposed into concentrically cylindrical sub domains. All the SPH particles, whose vector radius projection (in X,Y plane) lays between two consecutive cylinders, are assigned to the same process. A modified serial version of the code run on every sub domain. The interaction between particles in a sub domain with that owned by the two adjacent one is obtained defining "edge particles". All parameters of the "edge particles" are exchanged between two consecutive couple of process by, Message Passing Interface (MPI) standard. The sub-domain radius R_k are not fixed in order to ensure load balancing among processors. At each time step all internal SPH particles are processed and advanced after the density, pressure and gravitational forces are calculated considering the presence of the "edge particles". At the end of each time step particles migrating to the two consecutive sub domains are exchanged and the host particles are refreshed. The parallel code we have realized has run in a cluster of bi processors workstations linked by a fast local Internet link and on the CINECA (Bologna) IBM PC Cluster.

Key words. Numerical computation, Astrophysics

1. Introduction

The Smoothed Particle Hydrodynamics (SPH) technique is a grid free numerical method for simulating hydrodynamic phenomena. SPH is mainly used for astrophysical problems including star formation, supernova explosions, stellar collisions, galaxy formation and galactic interactions. It was first described in 1977 by Lucy, L.B. (1977) and in 1983 by Monaghan et al. (1983).

In SPH the matter distribution is divided into small overlapping mass packets, called particles, which do not exchange matter. The particles move in space following the motion of the fluid while interacting with their neigh-

Correspondence to: G. Gerardi Viale delle Scienze Ed. 18, 90128 Palermo

bor particles. The system of hydrodynamic equations, in lagrangian form, i. e., the Navier Stokes equation, the equation of continuity, and the energy equation, together with an equation of state (like that is in the following)

$$\frac{d\rho}{dt} = -\rho\nabla\cdot\mathbf{v} \tag{1}$$

$$\frac{d\mathbf{v}}{dt} = -\frac{1}{\rho}\nabla p \tag{2}$$

$$\frac{d\epsilon}{dt} = -\frac{p}{\rho}\nabla\cdot\mathbf{v} \tag{3}$$

$$\frac{p}{\rho^\gamma} = k\texttt{(or any other state eq.)} \tag{4}$$

form a system of *coupled* partial differential equations. Applying the SPH formalism, these are transformed into a set of ordinary differential equations (ODEs). These ODEs are subsequently integrated numerically using standard methods. The field quantity, $A(\mathbf{r})$ are smoothed as (see (5))

$$A(\mathbf{r}) = \int A(\mathbf{r}')W(|\mathbf{r}-\mathbf{r}'|, h)dV(\mathbf{r}')) \tag{5}$$

where $W(|\mathbf{r}-\mathbf{r}'|, h)$ is an interpolating kernel differentiable and normalized to unity i.e. with the proprieties.

$$\int W(|\mathbf{r}-\mathbf{r}'|, h)dV(\mathbf{r}-\mathbf{r}') = 1 \tag{6}$$

The smoothing length h is a measure of the (compact) support of the kernel function. A common choice for the kernel is a cubic spline (see, e. g., Monaghan, J.J. (1992)). The above integral are evaluated at the particle positions $\mathbf{r}_i$, and are approximated by sums.

$$A(\mathbf{r}_i) = \sum_{k=1}^{N} A(\mathbf{r}_k)W(|\mathbf{r}_i-\mathbf{r}_k|, h)dV(\mathbf{r}_k) \tag{7}$$

Spatial derivatives of field quantities at the particle positions can be approximated by

$$\nabla A(\mathbf{r}_i) = \sum_{k=1}^{N} m_k\frac{A_k}{\rho_k}\nabla W(|\mathbf{r}_i-\mathbf{r}_k|, h) \tag{8}$$

2. SPH Representation of the Hydrodynamic Equations

The particles follow the motion of the fluid ($d\mathbf{r}_i/dt = \mathbf{v}_i$) and do not exchange matter ($dm_i/dt = 0$). To the previous equations we must define, Π_{ik}, that is the artificial viscosity:

$$\Pi_{ik} = \frac{-\alpha\overline{c}_{ik}\mu_{ik} + \beta\mu_{ik}^2}{\overline{\rho}_{ik}} \ \texttt{if}\mathbf{v}_{ik}\cdot\mathbf{r}_{ik} < 0 \tag{9}$$

$$\Pi_{ik} = 0 \ \texttt{if} \ \mathbf{v}_{ik}\cdot\mathbf{r}_{ik} \geq 0$$

Where c is the speed of sound and μ is defined by

$$\mu_{ik} \equiv \frac{(\mathbf{v}_{ik}\cdot\mathbf{r}_{ik})h}{\mathbf{r}_{ik}^2 + \eta^2} \tag{10}$$

and $A_{ik} \equiv A_k - A_i$ and $\overline{A}_{ik} = (A_k + A_i)/2$

The parameters α and β should be near $\alpha = 1$ and $\beta = 2$ for best results see Monaghan, J.J. (1992).

The quantity η prevents singularities for $\mathbf{r}_{ik} \approx 0$. It should be chosen such that $\eta^2 = 0.01h^2$.

The hydrodynamic equations, adding the introduced viscous terms, have the following SPH formulation:

$$\begin{aligned}
\rho_i &= \sum_{k=1}^{N} m_k W_{ik} \\
\frac{d\rho_i}{dt} &= \sum_{k=1}^{N} m_k\mathbf{v}_{ik}\cdot\nabla_i W i_k \\
\frac{d\mathbf{v}_i}{dt} &= -\sum_{k=1}^{N} m_k(\frac{p_k}{\rho_k^2} + \frac{p_i}{\rho_i^2} + \Pi_{ik})\cdot\nabla_i W_{ik} \\
\frac{d\epsilon_i}{dt} &= -\sum_{k=1}^{N} m_k(\frac{p_k}{\rho_k^2} + \frac{p_i}{\rho_i^2} + \Pi_{ik})\mathbf{v}_{ik}\cdot\nabla_i W_{ik}
\end{aligned} \tag{11}$$

Where $\mathbf{v}_{ik} \equiv \mathbf{v}_k - \mathbf{v}_i$, $W_{ik} \equiv W(\mathbf{r}_{ik}) \equiv W(|\mathbf{r}_k - \mathbf{r}_i|, h)$ and ∇_i means the gradient with respect to the coordinates of the particle i.

Note that the **advantage** of using a kernel with compact support is the finite interaction range of the particles, i.e., only a particle's

neighbors within a distance of the smoothing length, $2h$, contribute to the above SPH sums. Because SPH is a gridfree method, complicated geometries are relatively easy to model. Moreover the local resolution of SPH can be enhanced considerably by allowing for a variable smoothing length.
The **disadvantage** of the method is that neighborhood relationships change with time and have to be reestablished every time the particle positions have changed.

3. Problem Integration

The right hand side of the eqs 11, can be calculated at the same time by using any suitable algorithm. Popular schemes are the leapfrog algorithm, predictor-corrector and Runge-Kutta methods.
In our Serial Code we use the predictor-corrector one and it is implemented following Monaghan, J.J. (1988):

3.1. *Time Stepping*

The value of the ODE's integration time step is controlled and evaluated before a new iteration is performed. The value is defined by Courant condition, the force terms, and viscous diffusion term (Monaghan 89).

3.2. *Finding Interactions, Link list definition*

In order to calculate the right hand side of the hydrodynamical equations in the SPH formalism one has to evaluate four SPH sums, the first and second to determine the density and its derivative at each particle position, the third and the fourth to calculate derivatives of the field quantities.

The direct approach to evaluate the SPH sums is clearly not reasonable because it leads to a N^2 algorithm, where N is the number of particles, an efficient way to search for all interacting particles is required.

As said before, only nearby particles may interact each other in the SPH model because of the compactness of the kernel i.e., is forced to zero for distances greater than $2h$. To identify the particles close to another generic particle we use a **link list/head-of-cell** method Monaghan, J.J. (1988)

3.3. *Finding Solutions*

Finding solutions to our problem is straightforward.

- *Initialization*
- **Start the loop**
- Make the predictor half step: Calculated provisional half step positions, velocities, etc..
- From the half step new particle positions calculate the total occupied parallellepiped volume for link list
- Define the grid and create the link list
- Calculate mid point new density
- Calculate mid point pressure force, volume forces, energy, etc.
- Perform the corrector phase
- Calculate the new step time value (Courant and etc. conditions)
- If not at the end: **Repeat the loop**

4. Parallel Implementation

The goal we want to obtain with our parallelization in threefold:

1. Execution speed gain nearly proportional to the number of processors;
2. Processing a number of SPH particles so high that doesn't fit a single computer;
3. Implementation of the parallel code in such a way to make future additions to the physics easier.

The first two items are essentially the same; if the particles number becomes very high we need more RAM and more processing speed because the number of the interacting particle is also increasing. The third goal comes from our need to port the new physics, put in the serial code, immediately into the parallel one.

4.1. Parallel System Hardware and Software

To realize more processing power we choose to use no special hardware but a cluster of workstation of standard PC. We choose PC motherboard with two processors, a very high speed (1 GB/s) Internet communication card and a switch for the local Internet LAN. Only one PC (the master of the cluster) has two communication cards to obtain the link to the external world.

Because we choose to use a processors pool with local memory, to exchange data, we use the well known standard: "*Message Passing Interface*" (MPI). Several implementations of this standard exist. We tested two implementations: LAM and MPICH. The last one has, besides the version running in UNIX like systems, LINUX, also a special version that runs in Windows Operating system. With both versions we have the possibility to define as many nodes as processors we have on the same motherboard. This possibility is very useful because communications between processors of the same motherboard don't use the communication card but internal bus. For the communications between internal processors the latency time and velocity throughput bottleneck are not present. Moreover there is the possibility to simulate a complete cluster even in a system with a single processor. It means that we can develop and test parallel codes without any access to a real cluster. From MPI standard it is possible to choose the communication type more appropriated to the actual problem among a variety of modes. All the functions are implemented with threads so that the non-blocking communications have low latency penalization.

4.2. Choosing Parallel Implementation Paradigm

From the previous enumerated goals and from the consideration that our physical problem (accretion disks around central star) has a space domain with some degree of central symmetry, it seems logical to divide the overall domain into sub domains having this type of symmetry.

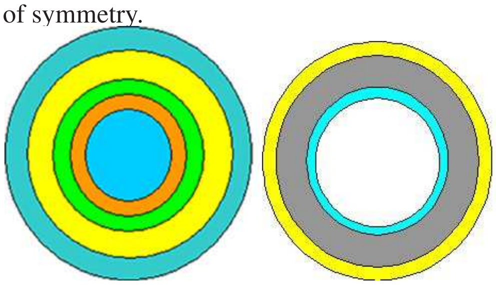


Fig. 1. Left panel: Horizontal cut view of domain decomposition in four sub domains. Right panel: Sub domain horizontal cut view with the inner and outer "edge regions".

In this way the code running in one sub domain is the same serial code (running in the total domain) with little modifications. The information exchanges are always between to consecutive domains. In other words the processing topology is of pipe type. The parallelization paradigm we use is the Multiple Instruction Multiple Data (MIMD) one. The three dimensional computational domain is decomposed into concentrically cylindrical sub domains with centers coincident with the coordinate origin. Figure 1a shows an horizontal cut view, in a case in which the domain is decomposed into four sub domains. All the SPH particles, having vector radius projection (in X,Y plane) lying between two consecutive cylinders, are assigned to the same sub domain. The number of sub domains is equal or multiple of the number of processors in the cluster.

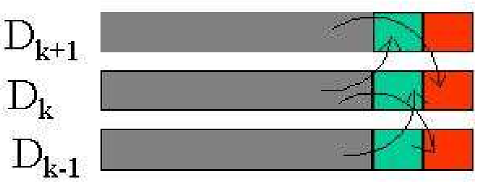


Fig. 2. Memory layout for particles data arrays. In the first *ni* positions we have data for internal sub domain particles, from $ni + 1$ to $ni + nbin$ data for particles of the inner edge region and from $ni + nbin + 1$ to $ni + nbin + nbout$ data for particles of the outer edge region.

From the previous sub-sections, we learned also that the SPH particles have a short-range interaction, $2h$. This means that particles, internal to a cylindrical sub domain, lying at a distance $\leq 2h$ from the limiting cylinders, have to interact with their analogues particles laying in the previous cylinder or in the next one. A particle internal to a sub domain, but lying in a $2h$ wide cylindrical region, is defined as "edge particle". We have two type of "edge particles": the ones interacting with the particles of the inner cylinder and the others interacting with the particles of the outer cylinder.

So that we extend the radius of each cylindrical sub domain dimension by $\pm 2h$. In this way a sub domain is divided into three regions: the first one is a "inner edge region" in common with the previous sub domain, the second one is its own region and the third one is an "outer edge region" in common with the next sub domain (see Figure 1b).

In order to have well ordered particles data in each processor and minimize latency time coming from communications of sparse data, all data of the particles are mapped into one matrix of vectors, each vector containing data for one particle. Data for edge particles are also copied in the same array but put after the data of the last internal particle of the sub domain. In more details: data for edge particles coming from the previous sub domain are written starting after the position occupied by the last internal particle and then data of the edge particle of the next sub domain are stacked on consecutive indices (see Figure 2). In any moment we have *ni* internal particles, *nbin* and *nbout* edge particles. In this way it is very simple to address data of particles of different types and transmit or receive them.
In each iteration loop we have to consider that:

1. The *nb* edge particles are important to supply to the *ni* particles the appropriate boundary conditions. This is achieved as follows: the *ni* particle are evolved taking into account the presence of the *nb* particles (for density, pressure forces etc. calculation), while the *nb* particles (in the guest domain) are evolved according to their original properties computed into their original sub domain. We observed that this procedure produces the minor perturbations.
2. At the end of the step time and before to start the next loop a check is performed to verify if some internal particle has migrated into the inner or into the outer sub domain. If so the particles data are exchanged.
3. Then, in each sub domain, the actual edge particles are identified and their parameters exchanged with the related sub domains.

4.3. *Parallel Time Step*

Due to different physical conditions in each sub domains, there is a high probability that each sub domain has a different stepping time, $\triangle t_k$ with $k = 1, 2, ..nd$. We want that each sub domain goes on with its own $\triangle t_k$, but, obviously, this fact requires a synchronization. To synchronize the domain times increments we make the following:

- Search the minimum of the $\triangle t_k$

$$\delta t_{min} = min(\triangle t_k) \qquad k = 1, ..nd$$

- For each sub domain the maximum time step, ts_k, we adopt is:

$$ts_k = Int(\tfrac{\triangle t_k}{\delta t_{min}}) \cdot \delta t_{min} \qquad k = 1, ..nd$$

  i.e. each ts_k has the value closest to its own $\triangle t_k$ conditioned to be an integer multiple of the δt_{min}.
- The time step for the entire domain "*Parallel time step*" will be

$$\delta t_{max} = \max(ts_k), \text{ for } k = 1, ..., nd.$$

  At δt_{max} all sub domains must find them selves synchronized.

The time evolution for the sub domain k to reach δt_{max} comes out through step increments equal to integer multiples of δt_{min} such that $n \cdot \delta t_{min} \leq ts_k$. In the TABLE 1, below, is shown an example how sub domains reach the synchronization. In this case we consider to have four sub domains and $ts_1 = 1 \cdot \delta t_{min}$; $ts_2 = 2{\cdot}\delta t_{min}$; $ts_3 = 3{\cdot}\delta t_{min}$; $ts_4 = 4{\cdot}\delta t_{min}$. The step advances follows this sequence. Note that, during each parallel stepping time, only a sub domain advances with a step time equal to the parallel one (δt_{max}) and only one sub domain advances the with a step time equal to minimum (δt_{min}). When synchronization has been reached, between adjacent sub domains, exchanging also edge particles data refresh the boundary conditions. In this way we accept an approximation in the evaluation deriving from boundary conditions on each sub domain: the properties of the hosted particles are fixed, while only their positions are updated. To minimize this approximation we impose the following limitation on the sub domain time step computation: *it must be less than the time needed to move any particles (in radial direction) by a distance equal to h*.

In order to avoid dead lock, before starting a new parallel step, the master of the cluster collect all $\triangle t_k$ from each sub domain, performs the calculation, we spoken about, and build a commands string for each sub domain, from witch all their next activity is planned, and dispatch it.

4.4. *Efficiency considerations*

Note that the first of the enumerated item of the sub-section 4.2 determine little overhead: construction of the link list take little time respect to other part of the code and no time is dedicated to calculation parameters new value of the edge particles.

The second item implies the searching and the transmiting/receiving data related to particles migrated from a sub domain to another.

The third item has the same type of implications but related to edge particles. Moreover, it needs to be execute after the end of the second item. The items (2) and (3) are present in the parallel code but not in the serial one. They determine the largest part of the overhead time. Some of it may be saved if we reduce at minimum the number of data for each particle to transfer.

5. Steady shock formation around Black Holes

The possibility of steady shock formation around black holes has been put forward by S. Chakrabarti in 1990 Chakrabarti (1990), see also Chakrabarti (1998). It has been successively confirmed by numerical simulation experiments by Chakrabarti and Molteni in 1993 Chakrabarti (1993) and following papers Chakrabarti (1998). This phenomenon has been shown to be very promising to explain several processes occurring in BH candidates Molteni (2001). We point to the fact that it can be also be considered a very critical test in the field of computational fluid dynamics. Indeed it is possible to obtain a plain analytical solution for a flow possessing a strong discontinuity with contemporary shear motion.

Let us resume the very basic physical ingredients of the phenomenon and, to make easy all the presentation, let us treat the 1D case in the cylindrical coordinates for an axially symmetric flow of gas. 1D means no vertical extension ($v_z = 0$, $Z = 0$) of the flow.

We adopt here

$$\Psi(r) = -\frac{GM_\star}{r - r_g} \tag{12}$$

the Paczyński & Wiita potential that mimics the Schwarzschild Black Hole force, where $r_g = \frac{2 \cdot G \cdot M}{c^2}$ is the Schwarzschild radius. Assume also the accreting gas is ideal (no viscosity and no cooling process is occurring). The angular momentum per unit mass of the flow, λ, must be conserved. In steady state regime we have also conservation of mass given by the equation:

$$\dot{M} = -r\rho v_r = const \tag{13}$$

In a conservative body force field with a potential $\Psi(r)$, also the pressure plays the role of a 'potential' and we have

$$\begin{aligned} \frac{1}{2}v(r)^2 + \epsilon(r) + \frac{P(r)}{\rho(r)} + \Psi(r) &= \\ \frac{1}{2}v(r)^2 + \frac{a(r)^2}{(\gamma-1)} + \Psi(r) &= B \end{aligned} \tag{14}$$

Here B is the Bernoulli constant (the thermal energy at infinity), a is the sound speed

$$a = \sqrt{\frac{\gamma \cdot P}{\rho}},$$

ϵ is the internal energy per unit mass, v is the speed of the flow and λ is the angular momentum, so that

$$\frac{1}{2}v^2 = \frac{1}{2}v_\phi^2 + \frac{1}{2}v_r^2 = \frac{\lambda^2}{2r^2} + \frac{1}{2}v_r^2. \tag{15}$$

If the entropy is conserved (no viscosity, no cooling, the presence of the shock will be taken into account few lines below) the polytropic relations are valid, $P/\rho^\gamma = P_0/\rho_0^\gamma$, so from this and from the speed of sound definition, we derive the following relation for density and sound speed

$$\rho = K \cdot a^{2/(\gamma-1)} \qquad \texttt{where} \qquad K = \rho_0/a_0^{\frac{2}{\gamma-1}}$$

So we have three unknown quantities ρ, v_r, a, and three equations. The solutions are functions of r with angular momentum and Bernoulli constant as parameters determining the specific shape of the solution.

Using the radial Mach number

$$m = -\frac{v_r}{a}$$

resolving a from the Bernoulli relation, and putting all terms in the continuity equation, we have the following implicit solution for the Mach number:

$$\dot{M} = r \cdot m \cdot a \cdot K \cdot a^{\frac{2}{\gamma-1}} \propto f(m) \cdot A\,(r, B, \lambda)$$

where f is function only of the mach number:

$$f\,(m) = \frac{m}{\left[\frac{m^2}{2} + \frac{1}{\gamma-1}\right]^{\frac{\gamma+1}{2(\gamma-1)}}}$$

and A is function only of r :

$$A\,(r, B, \lambda) = r\,[B - V\,(r, \lambda)]^{\frac{\gamma+1}{2(\gamma-1)}}$$

where

$$V\,(r, \lambda) = \frac{\lambda^2}{2r^2} - \frac{GM_\star}{r - r_g}$$

is the effective body forces potential (gravitational plus centrifugal).

For a given $\dot{M}$ this equation can be solved with any commercial symbolic software. The $f(m)$ function has a single maximum at $m = 1$, and it can be inverted both in the subsonic $0 \leq m < 1$ and supersonic $m > 1$ regions. The A function has in general two local minima at r_1 and r_2 with $r_1 < r_2$, which can be determined by solving the algebraic equation $dA/dr = 0$ numerically.

At large distances from the BH, the Mach number is $m \ll 1$, while close to the horizon it is $m \gg 1$. Since the $f(m) \cdot g(r)$ product must be constant along the flow, the maximum of $f(m)$ should be at one of the minima of $A(r)$, i.e. at r_1 or r_2. Therefore we can have two isentropic solutions $m_1(r)$ and $m_2(r)$

$$m_{1,2}(r) = f^{-1}\left[\frac{f(1)A_{\lambda,B}(r_{1,2})}{A_{\lambda,B}(r)}\right] \tag{16}$$

A standing shock can occur in the solution at r_{shock} if $m_2(r_{shock}) > 1$ and $m_1(r_{shock}) < 1$ are related by the Hugoniot relation

$$m_1(r) = \left[\frac{2 + (\gamma-1)m_2(r)^2}{2\gamma m_2(r)^2 - (\gamma-1)}\right]^{1/2} \tag{17}$$

First $m_2(r_{shock})$ can be determined by solving the algebraic equation

$$\frac{f(m_2)}{f[h(m_2)]} = \frac{A_{\lambda,B}(r_2)}{A_{\lambda,B}(r_1)} \tag{18}$$

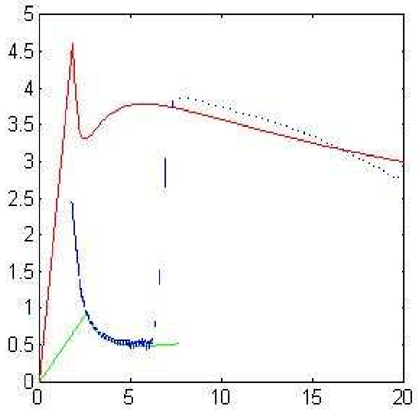
next the shock position $r_{shock} = A^{-1}_{\lambda,B}[f(1)A_{\lambda,B}(r_2)/m_2(r_{shock})]$ can be calculated. In general there can be two solutions, but only the outer one is stable.

One important point from the computational point of view is that for the pure 1D case the simulation experiment must reproduce the analytical result, if the code is good!

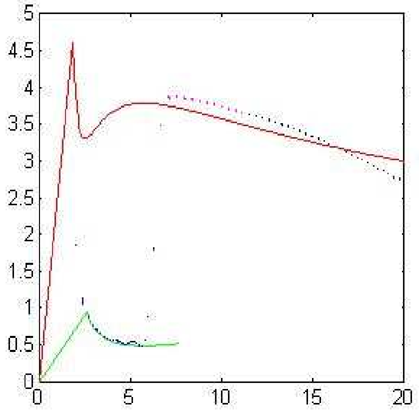
Note that this possibility is due to the absence of vertical motion. So also in 2D simulations in the XY plane, but with no Z motion it is also possible a direct comparison between analytical and numerical solutions.

In the last case shear motion is present in the flow since the gas is rotating in its fall towards the BH horizon.

Figure 3 shows a typical solution for $\gamma = 5/3$, $\lambda = 1.8$, $B = 0.001$. In order to check our code in this case we don't simulate real rotation of the gas .



(a) Serial Shock simulation results



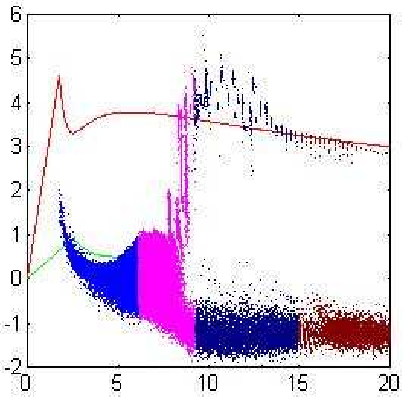
(b) Parallel Shock simulation results

Fig. 3. 1D Shock simulation with serial (a) and Parallel (b) code.
Simulations parameters are: $\gamma = 5/3$, ang. mom. = 1.8, $B = 0.001$

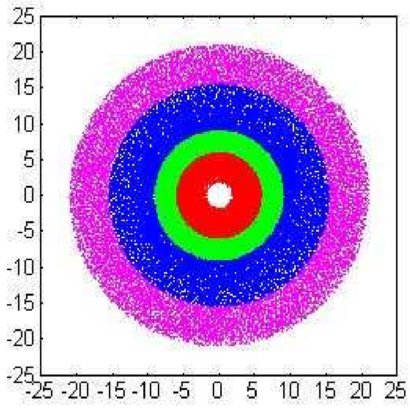
Note that playing with real rotation and fictious centrifugal forces we can obtain a variety of solutions with the same radial shock position, same radial Mach number, same density profile. The results of these simulations can be used to measure the numerical accuracy of any code: indeed the same shock profile must be obtained even with a mix of real and fictious rotation. The important point is that real angular momentum and the fictious force satisfy the equation

$$\lambda^2_{fict} = \lambda^2_{real} + sign(\lambda_{teor} - \lambda_{real})\lambda^2_{teor}$$

For example we could inject gas with keplerian rotation at the injection radius and put a fictious force equal to



(a) Serial Shock simulation results



(b) Parallel Shock simulation results

Fig. 4. 3D Shock simul. parallel code (a) shock pos. (b) XY plot of SPH Part.
Simulations parameters are: $\gamma = 5/3$, ang. mom. = 1.8, $B = 0.001$.
Different colors note different subdomain

$$\frac{\lambda^2_{fict}}{r^2} = \frac{\lambda^2_{real}}{r^2} - \frac{\lambda^2_{teor}}{r^2}$$

Sub Domain	δt_{min}	δt_{min}	δt_{min}	δt_{min}	NOTES
Sd1					*4 times* δt_{min}
Sd2					*1 time 2* δt_{min+}*2 times* δt_{min}
Sd3					*1 time 3* δt_{min+}*1 times* δt_{min}
Sd4					*1 time 4* δt_{min}

Step synchronization time. Vertical lines (in the middle of the table) mean that, at that time, each sub domain synchronizes with its adjacent ones. That is each sub domain looks for particles migrated into sub domain(s) (inner e/o outer) advanced for the same time and exchange data of migrated particles with it (them), then look for edge particles for the same sub domain(s) and exchange data of edge particles with it (them)

TABLE 1

It is clear that the faster rotational cases, due to the shear motion, may suffer of more numerical viscosity due to the intrinsic code properties: numerical accuracy, conservative structure, etc.. Figure 4 shows results obtained running the parallel code with the same input parameters but in this case the gas real rotation is active. We obtain same results with the serial code.

6. Conclusions

We studied the possibility of set up a geometrical domain decomposition and consequent parallelization of SPH code. It is shown that it is possible to achieve high CPU gain when the astrophysical problem does'nt involve self gravitation and has some degree of symmetry. Typical systems that can benefit of the proposed parallelization paradigm are accretion disks around a central black hole or binary systems with internal and external accretion disk. We also propose a physical problem that can be used as a test for any fluid dynamics code: the formation of standing shocks in the accretion flow of ideal gas onto a Black Hole.

It possible to derive the analytical profile of the solution of the fluid and compare it with the simulation one for different degree of rotation of the fluid. In this way it is possible to measure the numerical shear viscosity of the code.

Acknowledgements. Part of this work was supported by the ASI (Italian Space Agency) Contract number 0107949.

References

S.K. Chakrabarti, Theory of Transonic Astrophysical Flows, World Scientific, Singapore, 1990

S.K. Chakrabarti Observational Evidence for Black Holes in the Universe, Kluwer Academic, Dordrecht 1998

Chakrabarti, S. K. & Molteni D., Ap.J. 417 671 (1993).

Lucy, L.B. Astron. J. 1977 82, 1013.

Molteni, D.; Acharya, K.; Kuznetsov, O.; Bisikalo, D.; Chakrabarti, S. K. ,Ap.J. Letter563, L57,(2001).

Monaghan,J.J. Ann. Rev. Astron. Astrophysics 1992 30, 543

Monaghan,J.J. Computer Physics Communications 1988 48, 89

J.J. Monaghan & R.A. Gingold, Journal of Computational Physics 1983 52, 374