Large Scale Simulation of Particulate Flows

Ahmed H. Sameh and Vivek Sarin
Department of Computer Science
Purdue University
West Lafayette, IN 47907-1398
{sameh,sarin}@cs.purdue.edu

Abstract

Simulations of particles in fluid flows are of great interest to numerous industries using sedimentation, fluidization, lubricated transport, and hydraulic fracturing of hydrocarbon reservoirs. Simulating incompressible viscoelastic flows with millions of rigid particles is computationally a very challenging problem. In addition to using sophisticated modeling techniques and numerical algorithms, one must develop a scalable parallel formulation of the simulation. This task is further complicated by the dynamic nature of the system resulting from unrestricted motion of the particles. In this paper, we present an efficient algorithm for simulating particulate flows and discuss its parallel implementation. At each time step, a number of linear systems are solved using preconditioned iterative methods in which the matrix-vector product does not require explicit computation and storage of the matrix. The preconditioners developed for these systems are optimal so that convergence is assured in a fixed number of iterations. Moreover, these preconditioners do not require matrix inversion, and can be applied efficiently in parallel using their matrix-free forms. As a result, the algorithm is highly parallel and scalable, and achieves good speed improvement on the SGI Origin 2000.

1 Introduction

A number of important industrial problems can be successfully addressed with the ability to simulate particulate flows. To be of practical use, these simulations must be carried out for millions of particles in Newtonian as well as viscoelastic fluids. This will allow us to study the microstructural effects that produce clusters and anisotropic structures, analyse statistical behaviour, and derive engineering correlations for particulate flows.

Direct numerical simulation of particulate flows with millions of rigid particles is a very difficult task. Large scale simulations in a three dimensional domain are possible only on parallel computers. In addition to the difficulty associated with solving Navier-Stokes equations for incompressible fluids, we need to simulate motion of rigid bodies under the action of hydrodynamic forces and gravity.

We use a distributed Lagrange multiplier based fictitious domain approach (DLM) for these simulations (see, e.g., [7] for 2D particulate flows). DLM method is an alternative to Augmented Lagragian Eulerian (ALE) methods investigated in [4, 6, 8], and offers significant computational advantages. The operators can be implemented in a matrix-free form, and easily parallelized due to the use of uniform structured grids. An operator splitting method à la Marchuk-Yanenko is used for time discretization. This decouples each time step into three subproblems – the first one is the time-dependent Stokes problem, the second is a linearized advection-diffusion problem, and the third is a linearly constrained quadratic minimization problem that enforces rigid body motion on the particles. These subproblems are solved by the preconditioned conjugate gradients method.

The preconditioners developed for each of the subproblems are optimal, i.e., convergence is assured in a fixed number of iterations. In addition, these preconditioners possess the following desirable properties – they can be applied in a matrix-free form, they do not need to be stored, and they do not require matrix inversion. As a result, preconditioning can be implemented efficiently on multiprocessors. Some of these preconditioners are extensions of the ones proposed in [10, 9].

In this paper, we describe these preconditioners and outline their parallel implementation. We also discuss the parallelization of the DLM method. The next section describes the applications of particulate flow simulations. Section 3 presents the modeling techniques and Section 4 outlines the preconditioners for intermediate linear systems along with their parallel implementation. Experiments demonstrating good speed improvement on the SGI Origin 2000 are presented in Section 5. Conclusions are presented in Section 6.

2 Applications

Applications that stand to benefit from particulate flow simulations include sedimenting and fluidized suspensions, lubricated transport and hydraulic fracturing of hydrocarbon reservoirs. We now give an outline of these applications.

Sedimentation Columns and Fluidized Beds. Sedimentation columns and fluidized beds are used to suspend particles in a stream of fluid so that chemical reactions and heat transport can be enhanced considerably. This finds applications in diverse areas such as combustion of coal, coating of particulates, drying grains, and catalyzed conversion of light crudes to gasoline. The behavior of fluidized beds can be modified to avoid bubbling by using spouts and draft tubes provided one understands the dynamics of the particulate flow. Such studies are practical, and in some cases, possible only from direct simulation.

Lubricated Transport. Lubricated flows have the property that the low viscosity constituent migrates towards the walls, in effect greatly reducing the cost of transport of the high viscosity constituent. Coal slurries in water are examples of such a flow. Particulate flow simulations can provide insight into the nature of forces which push particles away from the wall. This information is necessary for exploiting the tendency to lubricate.

Hydraulic Fracturing. To increase the productivity of a hydrocarbon well, a slurry of sand in a highly viscoelastic fluid is pumped into the well. The resulting fracture in the pay zone is kept open by the sand, leading to increased productivity due to a higher-conductivity path through the sand for fluids to enter the well. Unfortunately, it is believed that the fluid flow into the well resembles lubricated transport, and interferes with a good vertical filling of the fractured reservoir. This in turn reduces well productivity, and can also interfere with the fracture growth process by blocking downward extension. Successful simulations of particulate flows can be used to develop efficient techniques for these activities.

3 Modeling Particulate Flows

Consider an incompressible fluid flowing in a channel, in which a pressure gradient is set against gravity (Fig. 1) and a number of solid particles are moving freely. The Reynolds number is large enough that inertia cannot be neglected. We are interested in determining the motion of both fluid and individual particles. The continuity and momentum equations

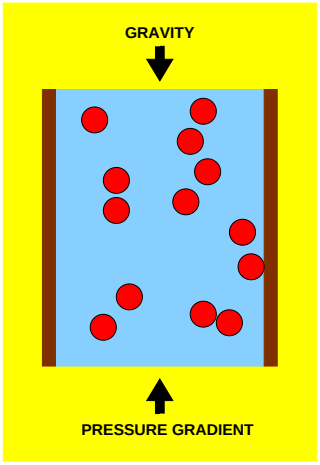

Figure 1. Particles moving in a channel.

for the fluid are given by:

$$\nabla \cdot \mathbf{u} = 0, \tag{1}$$

$$\rho \frac{\partial \mathbf{u}}{\partial t} + \rho \mathbf{u} \cdot \nabla \mathbf{u} = \rho \mathbf{g} - \nabla p + \nabla \cdot \tau, \tag{2}$$

where ρ is the fluid density, $\mathbf{g}$ is gravity, p is the pressure, and $\mathbf{u}$ is the fluid velocity. The tensor τ represents the extra-stress tensor; for a Newtonian fluid, it is given by

$$\tau = \nu \left(\nabla \mathbf{u} + \nabla \mathbf{u}^T \right). \tag{3}$$

The particles obey Newton's law:

$$\mathbf{M} \frac{d\mathbf{U}}{dt} = \mathbf{F}, \tag{4}$$

$$\frac{d\mathbf{X}}{dt} = \mathbf{U}, \tag{5}$$

where $\mathbf{X}$ and $\mathbf{U}$ are the generalized position and velocity vectors of the particles, respectively. The unknown vectors $\mathbf{X}$ and $\mathbf{U}$ incorporate both translational and angular components of the particle velocities. The matrix $\mathbf{M}$ denotes the generalized mass matrix and the vector $\mathbf{F}$ comprises of the forces and torques acting on the particles by the fluid as well as external fields such as gravity. Imposing a no-slip condition for the fluid on the surface of each particle, we obtain the following expression for fluid velocity at a point on the particle:

$$\mathbf{u} = \mathbf{U} + \mathbf{r} \times \Omega. \tag{6}$$

Here, $\mathbf{r}$ is the vector from the center of the particle to the point on its surface, and Ω is the angular velocity of the particle.

3.1 A Fictitious Domain Approach

Fictitious domain methods are used to reduce problems defined on complex domains to problems on simple shaped

domains containing the actual domain. Typically, one can use finite element discretization on uniform grids, and fast solvers for elliptic problems. For particulate flows, this offers the additional advantage of avoiding mesh generation at each time step. In contrast, one needs to generate a new mesh in ALE methods (see, e.g., [4, 6, 8]), whenever the mesh gets too distorted due to the motion of the particles. The fictitious domain method applied to particulate flows can be described as follows:

- fill each particle with the surrounding fluid,
- impose a rigid body motion to the fluid inside each particle, and
- relax the rigid body motion inside each particle by using a distributed Lagrange multiplier defined over the space region occupied by the particles.

Taking these steps, one arrives at the following generalized variational problem (see, e.g., [3] for details):

$$\begin{aligned}
&\rho \int_\Omega \frac{\partial \mathbf{u}}{\partial t} \cdot \mathbf{v} d\mathbf{x} + \rho \int_\Omega (\mathbf{u} \cdot \nabla)\mathbf{u} \cdot \mathbf{v} d\mathbf{x} \\
&- \int_\Omega p \nabla \cdot \mathbf{v} d\mathbf{x} + 2\nu \int_\Omega \mathbf{D}(\mathbf{u}) : \mathbf{D}(\mathbf{v}) d\mathbf{x} \\
&+ \int_\Omega \left(1 - \frac{\rho}{\rho_s}\right) \mathbf{M} \left(\frac{d\mathbf{U}}{dt} - \mathbf{g}\right) \cdot \mathbf{Y}
\end{aligned}$$

$$- < \lambda, \mathbf{v} - \mathbf{Y} \times \mathbf{r} > = \rho \int_\Omega \mathbf{g} \cdot \mathbf{v} d\mathbf{x} \quad (7)$$

$$\int_\Omega q \nabla \cdot \mathbf{u} = 0 \quad (8)$$

$$< \mu, \mathbf{u} - \mathbf{U} \times \mathbf{r} > = 0 \quad (9)$$

where $\mathbf{v}$, $\mathbf{Y}$, and q are basis functions for fluid velocity, particle velocity, and pressure, respectively, λ is the distributed Lagrange multiplier forcing rigid body motion on the fluid "filling" the particle, and μ is the corresponding basis function. The particle is assumed to have density ρ_s.

3.2 Operator Splitting for Time Discretization

Discretization in time is achieved via operator splitting à la Marchuk-Yanenko discussed in [5]. This decouples the three main difficulties in the equations, namely, incompressibility, nonlinearity, and rigid body motion. Each time step is subdivided into three parts, and a different subproblem is solved at each fractional time step. The first one is a time-dependent Stokes problem, the second is a linearized advection-diffusion problem, and the third is a linearly constrained quadratic minimization problem to enforce rigid body motion on the particles. These are described in more detail below.

1. Assuming $\mathbf{u}^n$, $\mathbf{U}^n$, and $\mathbf{X}^n$ are known at time n, solve the coupled equations

$$\rho \int_\Omega \frac{\mathbf{u}^{n+1/3} - \mathbf{u}^n}{\Delta t} \cdot \mathbf{v} d\mathbf{x} - \int_\Omega p^{n+1/3} \nabla \cdot \mathbf{v} d\mathbf{x} = 0, \quad (10)$$

$$\int_\Omega q \nabla \cdot \mathbf{u}^{n+1/3} d\mathbf{x} = 0. \quad (11)$$

2. Next, compute $\mathbf{u}^{n+2/3}$, $\mathbf{U}^{n+2/3}$, and $\mathbf{X}^{n+2/3}$ by solving

$$\rho \int_\Omega \frac{\mathbf{u}^{n+2/3} - \mathbf{u}^{n+1/3}}{\Delta t} \cdot \mathbf{v} d\mathbf{x} + \nu \int_\Omega \nabla \mathbf{u}^{n+2/3} \cdot \nabla \mathbf{v} d\mathbf{x}$$

$$+ \rho \int_\Omega (\mathbf{u}^{n+1/3} \cdot \nabla)\mathbf{u}^{n+2/3} \cdot \mathbf{v} d\mathbf{x} = \rho \int_\Omega \mathbf{g} \cdot \mathbf{v} d\mathbf{x}, \quad (12)$$

and assigning

$$\mathbf{U}^{n+2/3} = \mathbf{U}^n + \mathbf{g}\Delta t, \quad (13)$$

$$\mathbf{X}^{n+2/3} = \mathbf{X}^n + \frac{\Delta t}{2}(\mathbf{U}^n + \mathbf{U}^{n+2/3}). \quad (14)$$

3. Finally, compute $\mathbf{u}^{n+1}$, $\mathbf{U}^{n+1}$, $\mathbf{X}^{n+1}$, and λ^{n+1} via the solution of the coupled equations

$$\rho \int_\Omega \frac{\mathbf{u}^{n+1} - \mathbf{u}^{n+2/3}}{\Delta t} \cdot \mathbf{v} d\mathbf{x}$$

$$+ \left(1 - \frac{\rho}{\rho_s}\right) \mathbf{M} \left(\frac{\mathbf{U}^{n+1} - \mathbf{U}^{n+2/3}}{\Delta t}\right) \cdot \mathbf{Y}$$

$$- < \lambda^{n+1}, \mathbf{v} - \mathbf{Y} \times \mathbf{r} > = \rho \int_\Omega \mathbf{g} \cdot \mathbf{v} d\mathbf{x}, \quad (15)$$

$$< \mu, \mathbf{u}^{n+1} - \mathbf{U}^{n+1} \times \mathbf{r} > = 0, \quad (16)$$

and

$$\mathbf{X}^{n+1} = \mathbf{X}^n + \frac{\Delta t}{2}(\mathbf{U}^n + \mathbf{U}^{n+1}).$$

4 Parallel Implementation

In order to solve the coupled equations (10)–(11) for the time-dependent Stokes problem, we need to solve the following linear system:

$$\begin{pmatrix} \alpha I & B \\ B^T & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}, \quad (17)$$

where $\alpha = \rho/\Delta t$, and B^T is the discrete divergence operator. This is an indefinite saddle-point problem with potential to cause difficulty for iterative methods. The best approach is to solve the reduced linear system defined only on the pressure unknowns p, and then recover u from p. Specifically, we solve

$$B^T B p = B^T f, \quad (18)$$

and then compute $u = \alpha^{-1}(f - Bp)$. It is easy to show that the matrix $B^T B$ is spectrally equivalent to the discrete operator formed from $-\Delta$ with homogeneous Neumann boundary conditions (see also [1]). This implies that $-\Delta$ is an optimal preconditioner for the conjugate gradients method for solving (18), i.e., the iterations needed to converge to the solution would be independent of the mesh discretization. Since the preconditioner is also defined on a uniform grid (without particles), one can use a fast elliptic solver as a preconditioner. We use the iterative multilevel solver developed by Sarin and Sameh [9] that has been implemented in a matrix-free form for uniform grids.

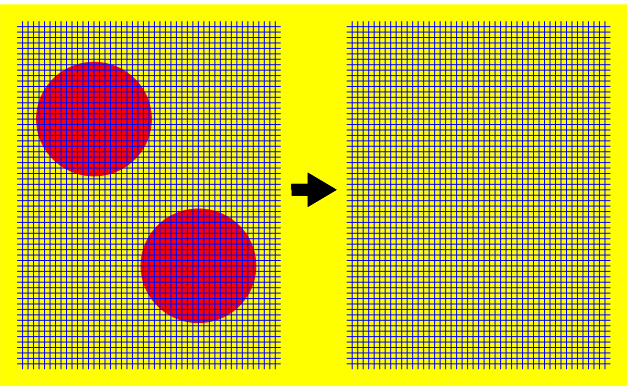

Figure 2. The first and second fractional time steps solve a linear and nonlinear problem, respectively, on a uniform grid. The particles are assumed to be filled with fluid, and thus, "invisible".

In the second fractional time step an advection-diffusion problem (12) is solved by a least squares/conjugate gradients algorithm [2] with two or three iterations. Each of these iterations requires solving a diagonally dominant sparse linear system obtained from the sum of $-\Delta$ with Dirichlet boundary conditions and a scaled identity matrix αI. In our experiments, these systems are solved with a few iterations of the Jacobi iterative method which can be implemented efficiently on parallel computers.

The third fractional time step (15)–(16) enforces rigid body motion on the particles by solving a linearly constrained quadratic minimization problem. Again, we need to solve a saddle-point sparse linear system:

$$\begin{pmatrix} \alpha I & & C_1 \\ & \beta \mathbf{M} & C_2 \\ C_1^T & C_2^T & 0 \end{pmatrix} \begin{pmatrix} u \\ U \\ \lambda \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ 0 \end{pmatrix}, \qquad (19)$$

where $\beta = \left(1 - \frac{\rho}{\rho_s}\right)/\Delta t$. As before, we solve the following reduced system for λ,

$$\left(\alpha^{-1} C_1^T C_1 + \beta^{-1} C_2^T \mathbf{M}^{-1} C_2\right) \lambda$$

$$= \alpha^{-1} C_1^T d_1 + C_2^T \mathbf{M}^{-1} d_2, \qquad (20)$$

and then obtain u and U. Finding an optimal preconditioner for this system is not straightforward; here we give a brief outline of how to construct the preconditioner. It is important to note that the number of unknowns λ for each particle is identical to the total number of mesh nodes that are either inside the particle or are connected to a node inside the particle. On the other hand, for a d dimensional simulation, there are at most $2d$ diagonal elements of $\mathbf{M}$ for each particle, i.e., the translational and angular velocities. Clearly, the system (20) is a low rank perturbation of a diagonal matrix, and can easily be inverted. In reality, however, $\alpha^{-1} C_1^T C_1$ is "almost" diagonal. Nevertheless, the diagonal of $\alpha^{-1} C_1^T C_1$ can be used along with the low rank perturbation from $\beta^{-1} C_2^T \mathbf{M}^{-1} C_2$ to obtain an optimal preconditioner for this system. Moreover, using Sherman-Morrison-Woodbury formula for the inverse, the preconditioning step can be made matrix-free and parallel.

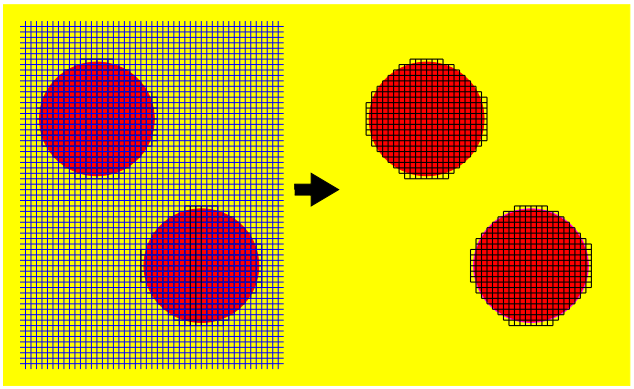

Figure 3. The third fractional time step enforces rigid body motion on that volume of fluid which is occupied by particles. Fluid "outside" the particles is unaffected at this step.

The fictitious domain approach has the advantage of using uniform grids. Fast parallel solution techniques for elliptic problems on uniform grids are well understood. Thus, mesh partitioning and load balancing issues in the first two fractional time steps are easily resolved. The third fractional time step can be parallelized effectively as well. The computation involves the unknown vector λ which is defined on a uniform grid that is restricted to each particle (see Fig. 3). In addition, this step involves projecting λ onto the fluid velocity near the surface of the particle, followed by projection of the updated fluid velocity onto λ. This is accomplished by product with the matrix $C_1^T C_1$. The outcome is that the fluid velocity array is referenced in the neighborhood of the particles. Unfortunately, this can lead to po-

tential load imbalance, especially when a disproportionate number of particles move into the subdomain allocated to a processor. Figure 4 illustrates this phenomenon for 240 sedimenting particles.

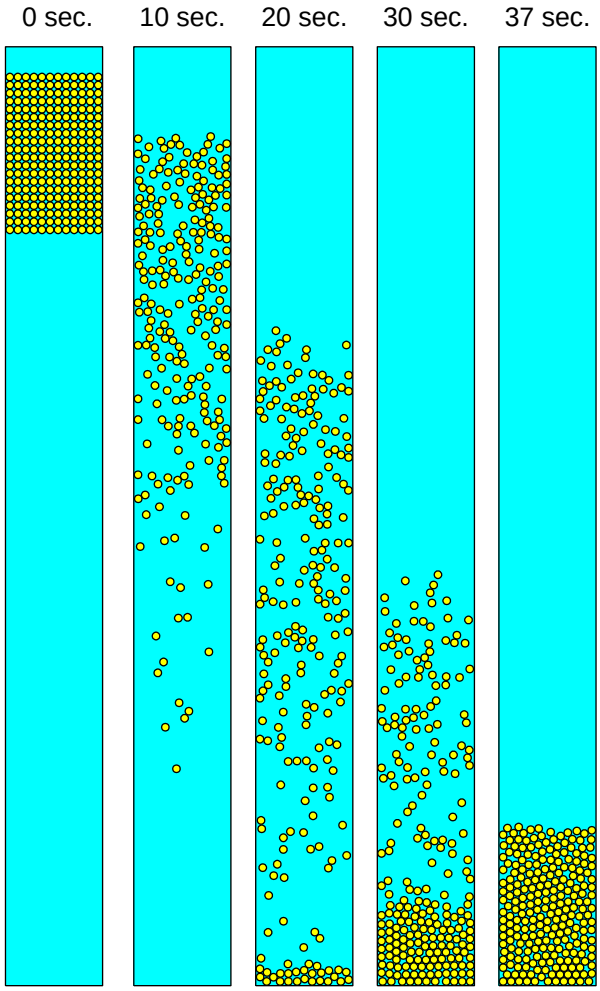

Figure 4. Sedimentation of particles in a channel.

This problem is addressed in the following way: particles are "handed" over to the processor that owns the subdomain containing the majority of the particle. A processor with disproportionately large number of particles can relinquish some particles to other lightly loaded processor. However, there is a tradeoff between load balance and the cost associated with communicating fluid velocity values on the particle boundary. The optimum solution may not be easy to find, and in fact, unnecessary if the time consumed by the third fractional step is much smaller than the first two.

5 Experiments

The parallel code for the algorithm described in the preceding sections has been used to simulate sedimentation and fluidization of particles in both two and three dimensions. We must point out that the parallelization effort for 3D code has not been completed; in fact, we anticipate much improved performance in the near future.

The 2D simulation consists of 240 circular particles of diameter 0.635 in a channel of height 40.64 and width 8.182. We assume the density of fluid, $\rho = 1.0$, density of particles, $\rho_s = 1.14$, and viscosity, $\nu = 0.01$. For sedimentation, the fluid velocity along the channel walls is $\mathbf{u} = \mathbf{0}$. For fluidization, $\mathbf{u} = \mathbf{0}$ along the the vertical walls, and along the horizontal walls, it is given by

$$\mathbf{u} = \mathbf{u}_{max}(1 - e^{-50t}),$$

where $\mathbf{u}_{max} = 0.25$. Initially, the particles are arranged so that their centers form a uniform 12×20 grid at the top of the channel for sedimentation experiments. This array is placed at the bottom in the case of fluidization. The time step is fixed at $\Delta t = 0.001$. The parallel performance of the code is shown in Table 1. Clearly, for a medium sized problem, our algorithm shows impressive speed improvement on a moderate number of processors.

Processors	Time (sec.)	Speedup	Efficiency
2	35.7	1.0	1.00
4	17.8	2.0	1.00
8	10.0	3.6	0.89
16	6.2	5.7	0.71

Table 1. Two dimensional simulation of 240 sedimenting particles. The time denotes average time taken per time step of the simulation on SGI Origin2000 with $h = 1/256$.

The 3D simulation consists of 21 spherical particles of diameter 0.635 in a channel of height 24.384, width 8.182, and depth 0.8128. The remaining parameters are identical to the 2D simulations. Table 2 presents the result of preliminary experiments on SGI Origin2000. Here, "Original" is used to denote the unpreconditioned serial code, S and S^* represents speed improvements in the overall code and the parallelized component, respectively, and S_{orig} denotes the net improvement in speed over the unpreconditioned serial code.

It is clear from the table that the parallel component of code shows impressive speed improvement. Even on a small problem, it shows a speed improvement of 10.4 on

Processors	Time (s)	S	S^*	S_{orig}
Original	179.0	—	—	—
1	93.8	1.0	1.0	1.9
2	55.2	1.7	1.8	3.2
4	30.7	3.1	3.5	5.8
8	19.2	4.9	6.3	9.3
16	13.6	6.9	10.4	13.2

Table 2. Three dimensional simulation of fluidization of 21 particles. The time denotes the time taken by 10 time steps on SGI Origin2000 with $h = 1/50$.

16 processors. We anticipate even better performance on larger problems. The only sequential component remaining is the solver for $-\Delta$ with homogeneous Neumann boundary conditions in the first fractional time step. This consumes approximately 10% of the sequential execution time, and is the main cause for speedup saturating to a constant value as the processors are increased. This is being remedied by replacing the solver with the parallel multilevel solver. Once parallelization is complete, we anticipate higher S for 3D simulations.

It is important to note that the overall parallel performance is affected by the preconditioners used at each fractional time step. The simulation proceeds fastest when using optimal preconditioners that may be somewhat restricted in parallelism. In this case, reasonable parallel speed improvement translates to enormous gains in the real time spent in these simulations. Clearly, this is much more favorable than achieving larger speed improvement at the cost of overall slowdown due to ineffective preconditioning.

6 Conclusions and Ongoing Work

We have developed an efficient parallel algorithm for simulating millions of particles in fluids. Our approach decouples the fluid equations from the particle equations by using distributed Lagrange multipliers to enforce rigid body motion for particles. This allows us to use fast parallel solvers for problems on uniform grids. We have also developed optimal preconditioners for various linear systems arising at each time step. These preconditioners do not need matrix inversion and have been implemented in matrix-free form. Our algorithm shows impressive speed improvement on the SGI Origin2000 for 2D as well as 3D simulations of sedimentation and fluidization benchmarks. In addition, we have improved upon the serial time by a large factor, mainly due to the use of optimal preconditioners and their matrix-free parallel implementation. Our approach also demonstrates that the choice of optimal algorithms with somewhat restricted parallelism is often preferable to highly parallel suboptimal algorithms. We anticipate that, after parallelizing remainder of the 3D code, our simulations will achieve very good parallel performance. This will enable us to tackle the Grand Challenge problems in particulate flows.

Acknowledgements

This work is supported by NSF HPCC Grand Challenge grant (ESC-95-27123). The authors wish to thank Tsorng-Whay Pan, University of Houston, for providing the serial code for particulate flow simulations, and Dan Joseph, University of Minnesota, for helpful discussions. The authors gratefully acknowledge the computational support provided by the National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign.

References

[1] J. Cahouet and J.-P. Chabard. Some fast 3d finite element solvers for the generalized Stokes problem. *Intl. J. Numer. Meth. in Fluids*, 8:869–895, 1988.

[2] R. Glowinski, Numerical methods for nonlinear variational problems. Springer-Verlag, New York, (1984).

[3] R. Glowinski, T.W. Pan, T.I. Hesla, D.D. Joseph, and J. Périaux. A distributed Lagrange multiplier/fictitious domain method for particulate flows. *Internat. J. of Multiphase Flow* (to appear).

[4] H.H. Hu, Direct simulation of flows of solid-liquid mixtures. *Internat. J. Multiphase Flow*, **22**, (1996), 335–352.

[5] G.I. Marchuk, Splitting and alternate direction methods. In *Handbook of Numerical Analysis*, **Vol. I**, (P.G. Ciarlet and J.L. Lions, eds.). North-Holland, Amsterdam, (1990), 197–462.

[6] B. Maury and R. Glowinski, Fluid particle flow: A symmetric formulation. C.R. Acad. Sci., Paris, t. 324, Série I, (1997), 1078–1084.

[7] T-W. Pan, V. Sarin, R. Glowinski, Ahmed H. Sameh, and J. Periaux. A fictitious domain method with distributed Lagrange multipliers for the numerical simulation of particulate flow and its parallel implementation. In *10th Parallel CFD Conference*, 1998.

[8] O. Pironneau, J. Liou, and T. Tezduyar. Characteristic-Galerkin and Galerkin least squares space-time formulations for advection-diffusion equations with time-

dependent domains. *Comp. Meth. Appl. Mech. Eng.*, **16**, (1992), 117–141.

[9] V. Sarin and A. Sameh. An efficient iterative method for the generalized Stokes problem. *SIAM J. Sci. Comput.*, **19**, (1998), 206–226, 206–226. 2, 335–352.

[10] V. Sarin. *Efficient Iterative Methods for Saddle Point Problems*. PhD thesis, University of Illinois, Urbana-Champaign, 1197.