Performance Metrics of a Parallel Three Dimensional Two-Phase DSMC Method for Particle-Laden Flows

Benzi John* and M. Damodaran**

Division of Thermal and Fluids Engineering,
School of Mechanical and Aerospace Engineering,
Nanyang Technological University,
50 Nanyang Avenue, Singapore 639798

Abstract

Parallelization aspects of a two-phase three dimensional Direct Simulation Monte Carlo (DSMC) method for modeling particle-laden gaseous flow in the vicinity of the Head-Disk Interface (HDI) gap in a modern Hard Disk Drive (HDD) enclosure are discussed in this work. Parallel performance metrics have been extracted and compared by executing this method on supercomputing platforms and various multi-core computing systems to illustrate the portability and scalability of the method. Super-linear speedup is obtained for the parallel case on all the computing platforms considered, implying a drastic reduction in the computational time.

1. Introduction

Multiphase flows which involve the transport of solid particles in a gaseous medium are found in a wide variety of applications. The vast majority of numerical models for studying the particle transport have been for cases where the continuum assumption is valid and tracking of particle trajectories in such models is achieved by coupling conventional CFD techniques with a suitable particle transport model as done in works such as Ali et al [1]. Particle transport modeling in rarefied flows has received considerable attention only recently with the advent of miniaturization and wide application of MEMS devices as in the hard disk drive and semiconductor industry. In a modern hard disk drive (HDD) the head-disk interface (HDI) gap size is of the order of a few nanometers which is much less than the mean free path of air. In such regions the flow falls in high Knudsen number regimes resulting in rarefied flow conditions and hence demands the use of a particle method like DSMC as in Huang et.al [2]. Multiphase models are required for the modeling of particle-laden flow and computation of particle trajectories that may enter the HDI region. Although the DSMC method is widely considered as the most reliable method for rarefied gas flow simulations, the application of DSMC is a computationally intensive method for realistic problems. The constraints associated with the DSMC method in terms of cell size, time step size and number of particles per cell makes DSMC a computationally intensive method, especially for the three-dimensional case. Parallel implementation of the DSMC method has been reported in literature such as by Dietrich and Boyd [3], Le Beau [4] and Benzi and Damodaran [5]. Two-phase modeling using the DSMC method for rarefied gas flows has received attention recently and parallelization aspects of the two-phase DSMC method have not been discussed in the literature to the best of our knowledge. In the current work a two-phase method is incorporated in the DSMC code to model particle-laden flow in the HDI gap. A detailed performance study is done on various computational platforms illustrating the performance of the parallel DSMC code.

* Graduate Research Student, E-mail: BENZ0001@ntu.edu.sg

** Associate Professor, E-mail: mdamodaran@ntu.edu.sg

2. Two-Phase DSMC Model

DSMC is a stochastic method introduced by Bird [6], in which a small number of computational particles are used to accurately capture the physics of a dilute gas governed by the Boltzmann equation. The two-phase model is based on computing forces acting on small particles in rarefied gas flow. The particle concentration is assumed to be dilute and the gas flow is not affected by the particle phase. For small particles locally free molecular flow conditions can be assumed if particle Knudsen number Kn_p is greater than one, where Kn_p is defined as $Kn_p = \lambda / R_p$, R_p being the particle radius. Gallis [7] provides expressions for computing the rates of momentum transfer to a solid particle from a single computational gas molecule under locally free molecular flow conditions. The total net force acting on the particle from all gas molecules with in the same DSMC cell as the particle can be computed by summing the individual contributions. The change in particle velocity and particle trajectory corresponding to the DSMC time step dt can then be computed based on the total net force acting on the particle. A two-phase DSMC model based on this approach has been incorporated in the DSMC method to compute particle trajectories in the HDI gap in the current work.

3. Parallel Implementation of Two-phase DSMC Method

The parallel two-phase DSMC program is implemented in C++ programming language with the aid of Message Passing Interface (MPI) [8] library routines. Various programming features such as c*lasses* and dynamic allocation of memory have been used in this work to enable efficient memory allocation and usage. In the parallel implementation the computational domain is partitioned uniformly along the *x* direction into several sub-domains as shown in Fig 1 and the mapped onto different processors. Uniform rectangular cells are considered for the collision cells to avoid any cell overlap between different processors and make it easier to sort particles and implement collisions. In a parallel DSMC simulation, data communication is associated with the transfer of molecules between adjacent sub-domains. All the non-contiguous pieces of information related to a gas molecule such as its spatial and velocity components are packed into a contiguous section of memory and transferred as a single message using the *MPI_Pack* and *MPI_Unpack* commands. For the two-phase DSMC calculations, communication needs arise if a particle crosses over to the next sub-domain after a certain number of time steps, in which case the new processor rank is computed. The index number of particle, the particle position and velocity has to be communicated so that the new processor does the particle trajectory computations.

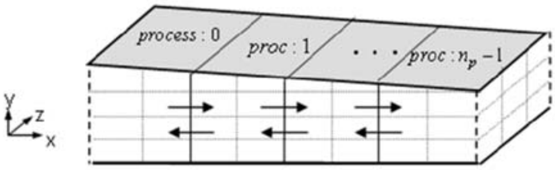


Figure 1: Schematic representation of the partitioning of the computational domain

To account for any parallel inefficiency arising from a load imbalance owing to a disparity in the number of computational molecules held between different processors, the domain is repartitioned once the system starts to attain a steady state, such that each processor has approximately the same number of gas molecules. Once steady state has been achieved, each processor does sampling for its own sub-domain. The results for both the flow variables and the particle trajectories from the two-phase model from each processor are sent to the host processor which gathers all the separate results into one single output file. The parallel two-phase DSMC approach is summarized in the flow chart shown in Fig 2.

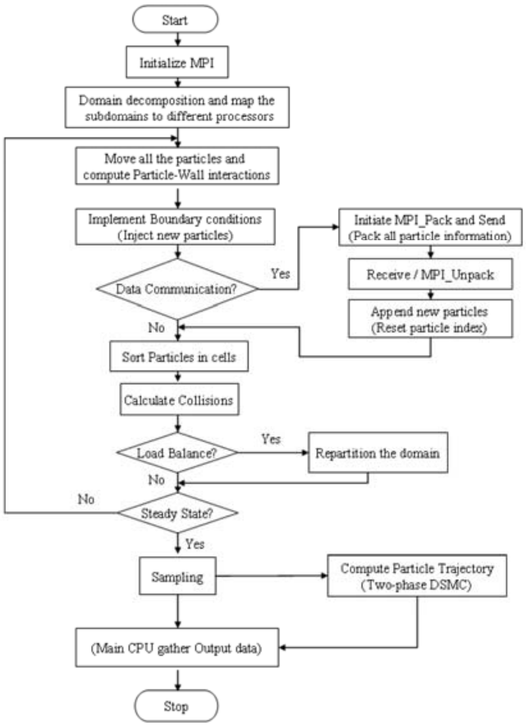

Start
Initialize MPI
Domain decomposition and map the subdomains to different processors
Move all the particles and compute Particle-Wall interactions
Implement Boundary conditions (Inject new particles)
Data Communication?
Yes
Initiate MPI_Pack and Send (Pack all particle information)
Receive / MPI_Unpack
Append new particles (Reset particle index)
No
Sort Particles in cells
Calculate Collisions
Load Balance?
Yes
Repartition the domain
No
No
Steady State?
Yes
Sampling
Compute Particle Trajectory (Two-phase DSMC)
(Main CPU gather Output data)
Stop

Figure 2: Flow chart for the parallel DSMC method

4. Parallel Performance Metrics

The scaling characteristics of the parallel three dimensional DSMC method is compared on two different supercomputing platforms, viz *Alpha Server (SC45), and SGI Origin 2000* by considering three cases with different number of particles, i.e. 20, 40 and 80 particles per cell. The *Alpha Server Supercomputer (SC45)* is a distributed memory platform having 44 nodes each of which is an SMP with four 1GHz processors having 1 GB memory and 8 MB cache. The *SGI Origin 2000* system is a shared memory platform consisting of 32 CPUs with 16 GB onboard memory, each utilizing processors of 250MHz and 4 MB cache. The performance metric, Speedup $S = T_s / T_p$, where T_s and T_p are the serial and parallel simulation times respectively is used in Fig. 3 to illustrate the scalability on the two platforms. The performance curves show good scaling characteristics with super-linear speedups obtained for all the three test cases. The super-linear speedup can be attributed to cache effects. By using multiple processors the size of sub-problem is reduced such that the data easily fits into caches resulting in significant reduction in memory access time. For a fixed load especially for the case of larger three dimensional DSMC problems, an increased reduction in CPU time can be obtained as the parallel program scales up well up to a higher number of processors. Interestingly the scalability characteristics of the parallel DSMC code with increase in computational load are different for the two platforms. For the SGI system which is a shared memory platform only a slight improvement in speedup is noted with increase in computational load. In contrast for the *Alpha Server (SC45)* system which is a distributed memory platform, significant improvement in performance is obtained with increase in number of particles. The best performance is observed for the case with 80 particles per cell wherein the super-linear speed is observed consistently till 64 processors. For the cases with lesser number of particles the speedup curve drops and falls below the theoretical speedup as the problem becomes communication bound for a larger number of processors.

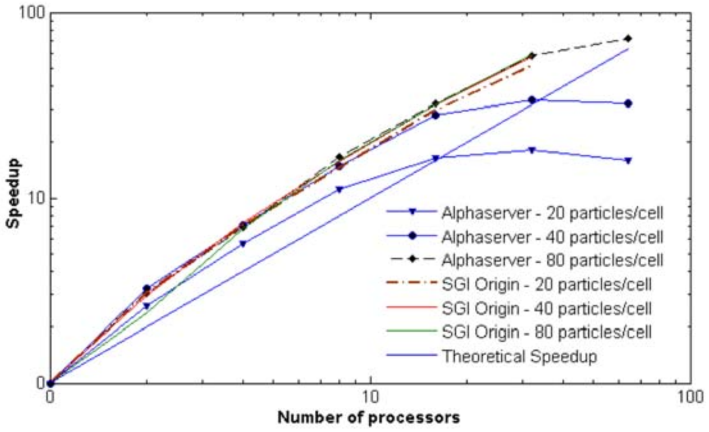


Figure 3: Performance curves for the parallel DSMC method

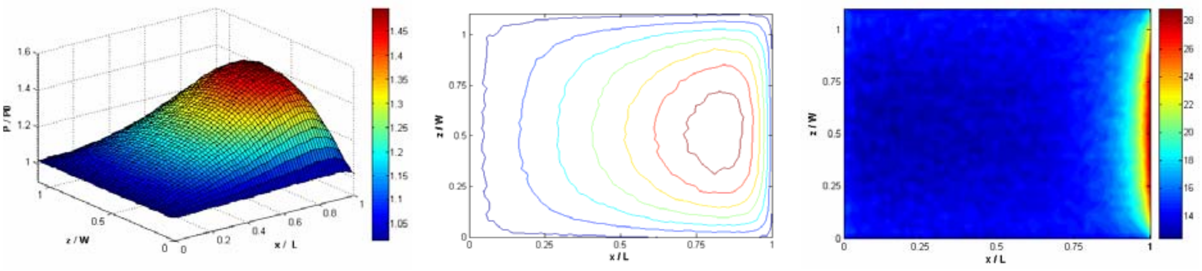
Parallel performance metrics have also been compared on three other platforms, viz. *Xeon PC Cluster*, *Sun SMP* and *IBM SMP* machine. *Xeon PC Cluster* is a distributed memory *system* with 9 nodes each having 4 GB memory and two dual-core *Xeon* processors of speed 2.33 GHz and 4 MB cache. The *Sun* and *IBM* systems are shared memory platforms. The *Sun SMP* system has 8 quad-core AMD Opteron processors of speed 2.3 GHz and 128 GB ram. The *IBM SMP* system has 4 dual-core processors of speed 4.2 GHz and 128 GB ram. The comparison of speedup from all the different computing platforms is summarized in Table 1. Super-linear speedup has been observed on all systems with multi-core processors. In particular significantly higher speedups are obtained with a higher number of processors for the quad core machine in comparison with the dual core machines. Since the processors have a significantly higher memory and processor speed, drastic reductions in computational time can be achieved using a fewer number of processors, as compared to the supercomputing platforms considered.

nproc	*Alphaserver SC-45*	*SGI Origin 2000*	*IBM SMP (Dual core)*	*Sun SMP (Quad core)*	*Xeon cluster (Dual core)*
2	3.21	3.06	3.09	3.09	3.17
4	7.17	7.04	8.97	7.37	7.34
8	14.8	14.67	15.35	17.48	8.15
16	28.1	29.63	27.59	34.93	10.79
32	33.6	51.68	37.17	64.13	
64	32.2				

Table 1: Summary of Speedup on different computing platforms

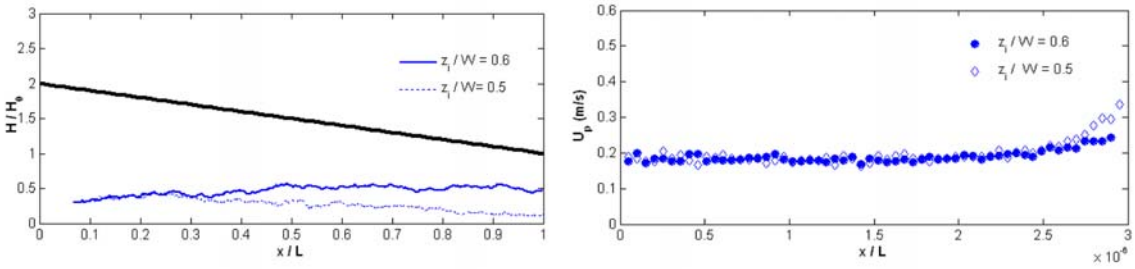
5 Computed Flow Results

Computed flow fields and particle trajectories in the HDI gap obtained using the parallel two-phase three dimensional DSMC code is shown in this section. The computed pressure profile in the HDI gap is shown in Fig 4(a). The contours of pressure and velocity on the disk surface are shown in Fig 4(b)-(c). Representative trajectories along the extent of the HDI gap followed by particles of radius $R_p = 5\,nm$ released from two locations along the width of the slider is shown in Fig 5(a) while the corresponding particle velocity is shown in Fig 5(b).



(a) (b) (c)

Figure 4. Computed flow fields in the Head-Disk Interface gap of (a) pressure profile (b)pressure contours and (c) velocity contours on disk surface



(a) (b)

Figure 5. Variation of (a) particle trajectory and (b) particle velocity along the extent of the HDI gap

6. Conclusion

The parallel performance of a three dimensional DSMC model has been investigated on both supercomputing platforms and various multi-core computing platforms. Super-linear speedup has been observed on all the computing platforms and this implies drastic reductions in computational time. The parallel DSMC code exhibits good scaling characteristics as the number of processors and computational load increase. The parallel performance metrics indicate that excellent speedup and a higher performance to cost ratio could be obtained with multi-core computing systems especially for large three dimensional problems.

Acknowledgment: *This work is part of a joint research project funded jointly by Seagate Technology International and Nanyang Technological University.*

References

[1] Ali, S., and Damodaran, M. “Computational Modeling of Particle Contamination in Turbulent Flow within a Hard Disk Drive Enclosure.” Progress in Computational Fluid Dynamics. Vol 8 No 6, pp 351-361, 2008.

[2] Huang, W., Bogy, D. B., and Garcia, A. L “Three-dimensional Direct Simulation Monte Carlo method for Slider Air Bearings.” Phys. Fluids, Vol 9, pp 1764–1769, 1997.

[3] DietRich, S., and Boyd, I. D. “Scalar and Parallel Optimized Implementation of the Direct Simulation Monte Carlo method.” JCP, Vol 126, pp 328-342, 1996.

[4] LeBeau, G. J. “A Parallel Implementation of the Direct Simulation Monte Carlo Method.” Comput. Methods Appl. Mech. Engg., Vol 174, pp 319-337, 1999.

[5] John, Benzi., and Damodaran, M. “Parallel Three-dimensional Direct Simulation Monte Carlo for Micro Flows.” Parallel Computational Fluid Dynamics 2007, Elsevier, North-Holland.

[6] Bird, G. A., “Molecular Gas Dynamics and the Direct Simulation of Gas Flows.” Oxford: Clarendon, 1994

[7] Gallis, M.A., Torczynski, J. R., and Rader, D. J. “An Approach for Simulating the Transport of Spherical Particles in a Rarefied Gas Flow via the Direct Simulation Monte Carlo Method.” Phys. Fluids. Vol 13 No 11, pp 3482-92, 2001.

[8] Gropp, William. “Using MPI : Portable Parallel Programming with the Message-passing Interface.” MIT press, 1994