Journal of Computational Physics 303 (2015) 455–469

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Adaptive kinetic-fluid solvers for heterogeneous computing architectures

Sergey Zabelok [a], Robert Arslanbekov [b], Vladimir Kolobov [b,*]

[a] Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Vavilova-40, Moscow, 119333, Russia
[b] CFD Research Corporation, Huntsville, AL 35805, USA

ARTICLE INFO

Article history:
Received 5 March 2015
Received in revised form 29 September 2015
Accepted 1 October 2015
Available online 13 October 2015

Keywords:
Unified Flow Solver
Adaptive Mesh Refinement
Discrete velocity method
Boltzmann kinetic equation
Direct Simulation Monte Carlo
Lattice Boltzmann Method
Graphics processing units
CUDA
MPI

ABSTRACT

We show feasibility and benefits of porting an adaptive multi-scale kinetic-fluid code to CPU–GPU systems. Challenges are due to the irregular data access for adaptive Cartesian mesh, vast difference of computational cost between kinetic and fluid cells, and desire to evenly load all CPUs and GPUs during grid adaptation and algorithm refinement. Our Unified Flow Solver (UFS) combines Adaptive Mesh Refinement (AMR) with automatic cell-by-cell selection of kinetic or fluid solvers based on continuum breakdown criteria. Using GPUs enables hybrid simulations of mixed rarefied-continuum flows with a million of Boltzmann cells each having a $24 \times 24 \times 24$ velocity mesh. We describe the implementation of CUDA kernels for three modules in UFS: the direct Boltzmann solver using the discrete velocity method (DVM), the Direct Simulation Monte Carlo (DSMC) solver, and a mesoscopic solver based on the Lattice Boltzmann Method (LBM), all using adaptive Cartesian mesh. Double digit speedups on single GPU and good scaling for multi-GPUs have been demonstrated.

1. Introduction

It is well known that transport phenomena can be described by either atomistic (kinetic) or continuum (fluid) models. Kinetic description in terms of particle distribution functions is more detailed and computationally more expensive compared to the continuum description in terms of density, mean velocity and temperature. Two methodologies have been used to solve the kinetic equations: statistical particle-based methods such as Direct Simulation Monte Carlo (DSMC) or Particle-in-Cell (PIC) [1,2] and direct numerical solutions using computational grid in phase space [3,4]. Continuum models are computationally efficient but have limited range of applicability. Mesoscopic models such as Lattice Boltzmann Method (LBM) attempt to bridge the gap between the two methods. Multi-scale kinetic-fluid models are being developed to enable using kinetic and fluid solvers in different parts of systems to achieve maximum fidelity and efficiency [5–10]. Adaptive kinetic-fluid models apply different solvers in dynamically selected regions of physical or phase space for efficient description of multi-scale phenomena in complex systems. Appropriate solvers are selected using sensors locally detecting phase space regions where kinetic approach is required and apply fluid models in other parts of the system.

For gas dynamics in mixed rarefied-continuum regimes, the adaptive kinetic-fluid approach has been first realized using an Adaptive Mesh *and* Algorithm Refinement (AMAR) methodology introduced for DSMC-fluid coupling [11]. Later, a Unified

* Corresponding author.
E-mail address: vik@cfdrc.com (V. Kolobov).

http://dx.doi.org/10.1016/j.jcp.2015.10.003
0021-9991/

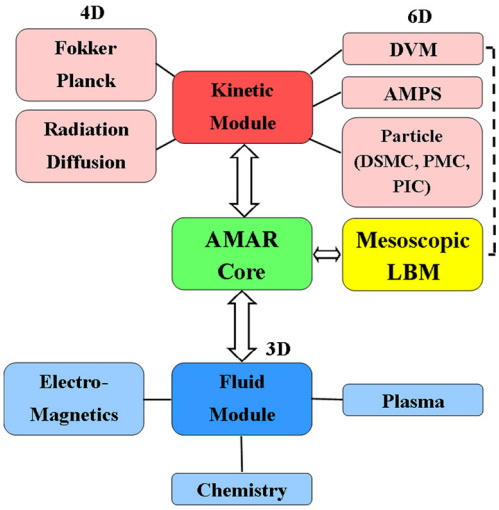

Fig. 1. The UFS AMAR framework.

Flow Solver (UFS) has been developed to combine Adaptive Mesh Refinement (AMR) with automatic cell-by-cell selection of direct Boltzmann solver or Euler–Navier–Stokes solvers [12] based on continuum breakdown criteria [5]. The AMAR methodology has been extended for hybrid modeling of radiation transport [13], and is now being developed for plasma simulations [6].

Fig. 1 shows the basic architecture of UFS. The AMAR core is implemented on top of Gerris Flow Solver (GFS) – an open source computing environment for solving partial differential equations with AMR [14]. GFS provides with automatic mesh generation for complex geometries, portable parallel support using the MPI library, dynamic load balancing, and parallel offline visualization. GFS physics includes time-dependent incompressible variable-density Euler, Stokes or Navier–Stokes equations with volume of fluid method for interfacial flows. A coarse-grained parallelization algorithm is based on "Forest of Trees" methodology [15].

UFS enables the higher degree of adaptation by using different physical models in different parts of the computational domain. The computational domain is decomposed into kinetic and fluid cells using physics-based continuum breakdown criteria. This methodology was first implemented for mixed rarefied-continuum flows and later extended for hybrid model of radiation transport coupling a Photon Monte Carlo (PMC) solver with a diffusion model of radiation transport selected based on the local photon mean free path [13].

The Kinetic Module in UFS can solve Boltzmann, Vlasov, and Fokker–Planck kinetic equations using different methods. Eulerian solvers use Discrete Velocity Method (DVM) for solving kinetic equations. The recently developed Adaptive Mesh in Phase Space (AMPS) methodology [16] can adapt mesh in both physical and velocity spaces. The DSMC, PMC and PIC modules are based on Lagrangian transport models [17]. The mesoscopic LBM solver uses a minimal set of discrete velocities as a subset of the DVM kinetic solvers [18].

Fluid Module in UFS contains multi-species Euler and Navier–Stokes solvers for reacting gas mixtures based on the Roe approximate Riemann solver, an exact Riemann solver with a Godunov-type scheme, AUSMPW+ scheme with MUSCL reconstruction, and the gas-kinetic schemes [12]. For plasma simulations, multi-temperature drift-diffusion models for electrons and ions coupled to Poisson solver for the electrostatic field are used [8].

The present paper demonstrates feasibility and benefits of adapting the UFS framework for heterogeneous computing architectures using Graphical Processing Units (GPUs). GPUs have become powerful computation accelerators for a wide range of devices from high-end supercomputers to battery-powered tablets, and laptops [19]. For programs that map well to GPU hardware, GPUs offer a substantial advantage over multicore CPUs in terms of performance, performance per dollar, performance per transistor, and energy efficiency.

Efforts required for porting existing software to CPU–GPU architectures depend on the problem type. GPUs can be very effective for regular programs that operate on large arrays or matrices and access them in statically predictable ways. These programs exhibit extensive data parallelism and access memory in a streaming fashion requiring little synchronization. Such *regular* codes, which require no significant change in program structure, have already been ported to GPUs [20]. Irregular codes associated with dynamic data structures (such as graphs, trees, etc.) are more difficult to port. Nevertheless, GPUs are capable of accelerating many irregular codes, and there is plenty of exploitable parallelism available even in irregular codes [21].

Expected acceleration from porting CFD solvers to GPUs depends on the type of model and the grid type [22]. Traditional CFD solvers are based on Navier–Stokes equations. Alternative methods based on LBM and gas-kinetic schemes utilize elements of the kinetic theory to provide solutions beyond the Navier–Stokes equations and thus often called mesoscopic methods. The LBM is both computationally expensive and memory demanding, but its explicit nature and the data locality