Toward live CFD-computing interaction and visualization using GPU acceleration

Florian De Vuyst, Christophe Labourdette, Christian Rey

Centre de Mathématiques et de leurs Applications CMLA, CNRS UMR 8536
et
NVIDIA CUDA Research Center ENS CACHAN

devuyyst@cmla.ens-cachan.fr
Objectives

- Get PDE approximate solutions instantaneously (« real time », co-simulation)

- Be able to directly act on the computations
Possible tracks ...

- Reduced-order modeling (POD, PGD, reduced basis method)
- High performance (parallel) computing
- Efficient algorithms, new paradigms (Lattice Boltzmann, ...)
- Multilevel modeling, surrogates
- Parallel computing on workstations (GPU, coprocessors)
Toward GPU computing for PDE pbs – Outline

1. GPU hardware architecture
2. Performance drivers
3. Focus on Lattice Boltzmann Methods (LBM)
4. Real time flow interaction
5. Work in progress: LB thermal-fluid Boussinesq system
6. Suspension flows

Perspectives
1. GPU hardware architecture
nVIDIA GPU FERMI compute architecture

(2*16)*16 = 512 cores

(courtesy of Nvidia)
NVIDIA – KEPLER family (2013)

NVIDIA GeForce GTX 690 (game)
- 2 x 1536 cores, 300 W
- 8 streaming multiprocessors (SM)
- Memory bandwidth 192 GB/sec
- MEM 4 GB (2048 MB per GPU)
- DRAM bus memory 512-bit GDDR5
- 2x1.8 Tflops SP, 2x130 Gflops DP

NVIDIA TESLA K20X (HPC)
- **2688 cores**, 250W !
- 14 streaming multiprocessors (SM)
- Memory bandwidth 250 GB/sec
- MEM 6 GB
- 3.95 Tflops SP, **1.31 Tflops DP** !
2. GPU Performance drivers
Performance drivers

- **Multiprocessor occupancy**
- **Byte-per-flop** ratio (mem bandwidth vs FP operations)
- **Memory** management: registers, cache, coalesced read/write memory, fixed neighboring patterns reads/writes
- **Warp divergence**: be careful to trees of conditional branches
- **Communication**: host-to-device **PCIe** bus bottleneck

[Williams et al., *Roofline*: an insightful visual performance model for multicore architectures, Com. ACM, 2009]
Questions for the design of numerical methods:

- **Explicit** vs implicit?
- **Cartesian** vs unstructured grid?
- Order of accuracy vs grid size?
- Specific vs undifferentiated treatment? (interfaces, ...)
- Boundary conditions: **embedded** strategy?
- How to achieve **condensed stencil**?
- Operator splitting, **alternating** directions
- Change the **model description**?
- Replace spatial derivatives by time derivative, particle derivative? ...
3. Focus on Lattice Boltzmann methods (LBM)
Lattice BGK (LBGK) model

Based on a simple discretization of the Boltzmann equation with BGK approximation for the collision term:

\[
f(x + ce_i \Delta t, e_i, t + \Delta t) - f(x, e_i, t) = \frac{1}{\tau} \left[ f^eq_i(\rho, u) - f(x, e_i, t) \right]
\]

\(c = \frac{\Delta x}{\Delta t}\).

\(e_0 = 0, \ e_1 = (1, 0), \ e_2 = (1, 1), \ldots\)

**Moments:**

\[
\sum_{i \in S} f(x, e_i, t) = \rho(x, t),
\]

\[
\sum_{i \in S} ce_i f(x, e_i, t) = \rho u(x, t).
\]
Lattice D2Q9 BGK model

- **D2Q9 unknowns**
  \[ f_i(x, t) \equiv f(x, ce_i, t), \ i \in \{0, \ldots, 8\}. \]

- **LB equation**
  \[
  f_i(x + ce_i, t + \Delta t) = f_i(x, t) + \omega \left[ f_i^{eq}(\rho(x, t), u(x, t)) - f_i(x, t) \right]
  \]

- **Requirements**
  \[
  \omega = \frac{1}{\tau}
  \]
  \[
  \sum_{i \in S} f_i^{eq}(\rho, u) = \rho, \\
  \sum_{i \in S} ce_i f_i^{eq}(\rho, u) = \rho u, \\
  \sum_{i \in S} c^2 e_i \otimes e_i f_i^{eq}(\rho, u) = \rho u \otimes u + pI
  \]
  + some discrete symmetry/invariance conditions
Practical implementation « stream-and-collide »

1. **Streaming step**

\[ \tilde{f}_i(x, t) = f_i(x + ce_i, t) \]

2. **Collision step**

- Compute the moments \( (\rho, \rho u)(x, t) = \sum_{i \in S} (1, ce_i) \tilde{f}_i(x, t) \)

- Compute the equilibrium function \( \tilde{f}_i^{eq} = \rho w_i \ldots \)

- Time advance

\[ f_i(x, t + \Delta t) = \tilde{f}_i(x, t) + \omega \left[ \tilde{f}_i^{eq}(\rho(x, t), u(x, t)) - \tilde{f}_i(x, t) \right] \]

- Explicit method
- Underlying cartesian grid
- Fixed (small) stencil
- Elementary operations
- Very easy to implement

Particularly suitable for GPU computing
4. Real time visualization & flow interaction
Visualization & interaction

• Direct GPU compute/visualization binding (Pixel Buffer Object PBO) + OpenGL

• Dynamic mask array for adding/removing wall BC

• IR U-Pointer device for large screen interaction (seminars, conferences)

• Android tablets for controls & GUI. Use of OSC for bidirectional communication between GPU-workstation and tablet (coll. K. Labourdette).
5. Work in progress: LB thermal-fluid Boussinesq system
Thermal + CFD coupling : Boussinesq model

\[ \nabla \cdot \mathbf{u} = 0, \]

\[ \partial_t \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p - \frac{\mu}{\rho_0} \Delta \mathbf{u} = \mathbf{g} \left( 1 - \beta (T - T_0) \right), \]

\[ \partial_t T + \mathbf{u} \cdot \nabla T - \nabla \cdot (\kappa \nabla T) = 0, \]

\[ + IC + BC. \]

\[ \mu, \alpha, \kappa > 0. \]

Convection, diffusion, source term, coupling.
2D Lattice Boltzmann discretization

- D2Q9 LBGK lattice for fluid
- D2Q4 LBGK lattice for the thermal equation

Half-discretization (continuous in time):

\[ \partial_t f_i + c e_i \cdot \nabla_x f_i = \frac{f_i^{eq} - f_i}{\tau \Delta t} - \frac{e_i \cdot g}{2c} (1 - \beta(T - T_0)) \]

\[ (\rho, u) = \sum_{i=0}^{8} (1, e_i) f_i, \quad f_i^{eq} = f_i^{eq}(\rho, u) \]

\[ \partial_t k_i + c e_i' \cdot \nabla_x k_i = \frac{k_i^{eq} - k_i}{\tau' \Delta t} \]

\[ T = \sum_{i=1}^{4} k_i, \quad k_i^{eq} = \frac{T}{4} \left(1 + 2 \frac{u \cdot e_i'}{c} \right). \]
Validation of the method

\[ g = 9.81, \quad Re = 2300, \quad Pr = 0.71, \quad Ra = 3 \times 10^5, \quad \nu = Re^{-1}, \quad \kappa = \frac{\nu}{Pr}, \quad \beta = \frac{Ra \nu \kappa}{g H^3 \Delta T} \]

Collaborators: S. Faure, L. Gouarin, B. Graille
(U. Paris-Sud Orsay)
Forseen application: solar powered air conditioning

Design the chimney shape in order to reduce recirculating zones at the top and maximize the flow rate

(see [Chenier et al.], [Le Quéré, Sergent & co-workers] on the subject)
6. Suspension / sediment gravity flows
Suspension flows

Conservative coupling

\[ \frac{dx_p}{dt} = u_p, \]

\[ m_p \frac{du_p}{dt} = \int_{\partial S_p} \sum \nu \, d\sigma. \]

Boundary conditions, flux formulation

[Dubois, Lallemand 2008]

Exact trajectory approximated by a broken-line lattice path (random choice stochastic method)

\[ (x_p, m_p) \]

\[ \Rightarrow \text{Geometric elements and intersecting nodes computed once-for-all} \]
Perspectives

- Euler-Euler multiphase flows (suspension)

- Immiscible fluids with moving interfaces

- NVIDIA outlook: 20 Tflops DP/GPU by 2018 → real-time interactive 3D?

- CPU-GPU convergence (be aware) ...
Acknowledgments

- NVIDIA CUDA Research Center ENS Cachan

- Institut fédératif FARMAN ENS Cachan

- Labex LMH Fondation Mathématique Hadamard FMJH