Large-scale stencil and particle applications and their performances on a GPU supercomputer

Takayuki Aoki

Global Scientific Information and Computing Center
Tokyo Institute of Technology
Bottlenecks

- Communication (memory) bandwidth
- Memory Hierarchy

---

**Large-scale GPU Applications**

I. Weather prediction
II. Aerodynamics by Lattice Boltzmann Method
III. Two-Phase Flows
IV. Granular material Simulation
V. Phase-Field Simulation for Dendrite Solidification

- A Performance Model
Weather Prediction

Collaboration: Japan Meteorological Agency (JMA)

Meso-scale Atmosphere Model:

Cloud Resolving Non-hydrostatic model
Compressible equation taking consideration of sound waves.

**Meso-scale**

2000 km Typhoon, hurricane

a few km Tornado, Down burst Heavy Rain

Atmosphere Model

**Dynamical Process:**

Full 3-D Navier-Stokes Equation

\[
\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\frac{1}{\rho} \nabla P - 2 \Omega \times \mathbf{u} - \Omega \times (\Omega \times \mathbf{r}) + \mathbf{g} + \mathbf{F}
\]

**Physical Process:**

Cloud Physics, Moist, Solar Radiation, Condensation, Latent heat release, Chemical Process, Boundary Layer

“Parameterization” including sin, cos, exp, …in empirical rules.
Full GPU Implementation: ASUCA

ASUCA Production Code
✓ A next-generation high resolution weather simulation code that is being developed by Japan Meteorological Agency (JMA)
✓ ASUCA succeeds the JMA-NHM as an operational non-hydrostatic regional model at JMA

J. Ishida, C. Muroi, K. Kawano, Y. Kitamura, Development of a new nonhydrostatic model “ASUCA” at JMA, CAS/JSC WGNE Research Activities in Atmospheric and Oceanic Modelling.

Entire Porting Fortran to CUDA

• Rewrite from Scratch

Fortran
C/C++
CUDA

Original code at JMA
Changing array order
GPU code

✓ x, y, z (i, k, j)-ordering
✓ x, z, y (i, k, j)-ordering
✓ x, z, y (i, k, j)-ordering

1 Year
Introducing many optimizations, overlapping the computation with the communication, kernel fuse, re-ordering kernel, ...
M2050 (Fermi core) 1 GPU

Multi-GPU : Domain decomposition

Copyright © Global Scientific Information and Computing Center, Tokyo Institute of Technology
Overlapping between Computation and Communication

Asynchronous Data transfer

SendBuffer → Sync → MPI → RecBuffer

Stream 1

Asynchronous Data transfer

Stream 2

TSUBAME 2.0 Weak Scaling

145.0 Tflops
Single precision
76.1 Tflops
Double precision
Fermi core Tesla M2050
3990 GPU

SC’10 Technical Paper
Best Student Award finalist
ASUCA Typhoon Simulation
5km-horizontal resolution  $479 \times 466 \times 48$

ASUCA Typhoon Simulation
500m-horizontal resolution  $4792 \times 4696 \times 48$
Using 437 GPUs
Problems with Accelerator Programming

1. Accelerators needed to get ready for Exascale
2. Existing applications often not written with accelerators in mind:
   a. Fixed storage order
      (GPGPUs need warp-first)
   b. Coarse-grained parallelism
      (Accelerators need fine-grained)

<table>
<thead>
<tr>
<th>Approach</th>
<th>Examples</th>
<th>Pros</th>
<th>Cons</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libraries</td>
<td>ArrayFire, BLAS</td>
<td>Optimization comes 'for free'</td>
<td>Limited control, limited use</td>
<td>High portation effort</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Relearning of practices</td>
</tr>
<tr>
<td>Directives</td>
<td>OpenMP, OpenACC</td>
<td>Generally applicable</td>
<td>No satisfying standard to solve 2.a, 2.b</td>
<td>High portation effort</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Relearning of practices</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>No performance portability</td>
</tr>
</tbody>
</table>

Hybrid Fortran framework

1. New set of directives to support flexible storage- and loop orders (solve 2.a, 2.b).
2. Goal: Unified codebase, performant on CPUs and accelerators

Code transformation with different backend implementations for different architectures and kernels.

```
subroutine wrapper(a, b, c)
    real, dimension(NZ), intent(in) :: a, b
    real, dimension(NZ), intent(out) :: c
    !domainDependent(domName(x,y), domSize(NX,NY))
    a, b, c
    !send domainDependent
    #parallelRegion appliesTo(CPU) domName(x,y), domSize(NX,NY))
    call add(a, b, c)
    !send parallelRegion
    end subroutine

subroutine add(a, b, c)
    real, dimension(NZ), intent(in) :: a, b
    real, dimension(NZ), intent(out) :: c
    integer :: x
    !domainDependent(domName(x,y), domSize(NX,NY))
    a, b, c
    !send domainDependent
    #parallelRegion appliesTo(GPU) domName(x,y), domSize(NX,NY))
    a(x) + b(x)
    !parallelRegion
end subroutine
```
Lattice Boltzmann Method

\[ \frac{\partial f_i}{\partial t} + \mathbf{e}_i \cdot \nabla f_i = -\frac{1}{\lambda} (f_i - f_{i}^{eq}) \]

\[ f_{i}^{eq} = \rho w_i \left[ 1 + \frac{3}{c^2} (\mathbf{e}_i \cdot \mathbf{u}) + \frac{9}{2c^4} (\mathbf{e}_i \cdot \mathbf{u})^2 - \frac{3}{2c^2} (\mathbf{u} \cdot \mathbf{u}) \right] \]

Strongly Memory Bound Problem:

**Collision step:**

**Streaming step:**

- \( i \) is the value in the direction of \( ith \) discrete velocity
- \( \mathbf{e}_i \) is the discrete velocity set;
- \( w_i \) is the weighting factor
- \( c \) is the particle velocity
- \( \mathbf{u} \) is the macroscopic velocity
Classification of Turbulence Model

<table>
<thead>
<tr>
<th>Method</th>
<th>Computational Cost</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNS</td>
<td>Only molecular viscosity</td>
<td>Inacceptable high: Very small area</td>
</tr>
<tr>
<td>LES</td>
<td>Solving equation on grid and modeling sub-grid</td>
<td>acceptable</td>
</tr>
<tr>
<td>RANS</td>
<td>Reynolds average</td>
<td>small</td>
</tr>
</tbody>
</table>

LES (Large-Eddy Simulation)

\[ f_i(x + c_i \Delta t, t + \Delta t) = f_i(x, t) - \frac{1}{\tau_*}(f_i(x, t) - f_i^{eq}(x, t)) + F_i \]

Relaxation time for LES model

\[ \tau_* = \frac{1}{2} + \frac{3\nu_*}{r^2 \Delta t} \]

\[ \nu_* = \nu_0 + \nu_t \]

Molecular viscosity and Eddy viscosity

* Coherent-Structure Smagorinsky model

Local determination of model coefficient

Energy spectrum
LES modeling

Smagorinsky model

\[ \tau_{ij} = -2 \nu_{SGS} S_{ij} \]
\[ \nu_{SGS} = C \Delta^2 |S| \quad C: \text{const} \]

- Simple
- \( \Delta \) inaccurate for the flow with wall boundary
- \( \Delta \) empirical tuning for the constant model coefficient

Dynamic Smagorinsky model

\[ \nu_{SGS} = C \Delta^2 |S| \]
\[ C = \frac{L_{ij} L_{ij}}{M_{ij} M_{ij}} \]
\[ M_{ij} = 2 \Delta^2 |S| S_{ij} - 2 \Delta^2 |S| S_{ij} \]

- \( \Delta \) applicable to wall boundary
- \( \Delta \) complicated calculation
- \( \Delta \) average process over the wide area
- \( \rightarrow \) not available for complex shaped body
- \( \rightarrow \) not suitable for large-scale problem

Coherent-Structure Smagorinsky model

\[ \nu_{SGS} = C \Delta^2 |S| \quad \rightarrow \quad \text{model coefficient determined by the second invariant of the velocity gradient tensor} \]
\[ C = C_1 |F_{CS}|^{3/2} \]
\[ F_{CS} = \frac{Q}{E} \]
\[ Q = -\frac{1}{2} \frac{\partial u_i}{\partial x_i} \frac{\partial u_i}{\partial x_j} \quad E = -\frac{1}{2} \left( \frac{\partial u_i}{\partial x_i} \right)^2 \]
\[ (-1 < F_{CS} < 1) \]

*H. Kobayashi, Phys. Fluids. 17, (2005).*

Turbulence Statistics

768x256x384, 1536x512x768
Computational Area

Major part of Tokyo
Including Shinjuku-ku, Chiyoda-ku, Minato-ku, Meguro-ku, Chuou-ku,

10km × 10km

Building Data:
Pasco Co. Ltd.
TDM 3D
Performance Evaluation via roofline model

Performance estimation by using the Roofline model

CUDA Tuning
* Using SFU (Super Function Unit) and single precision computation
* Kernel fusion of the collision step and streaming step
* Loop unrolling for optimization

+ Reduction of the address calculation by 32-bit compiling

Performance = \frac{\text{FLOP/Byte} \times F}{\text{FLOP/Byte} + \frac{F}{B}}

F = 1030 \text{ GFLOPS}
B = 148 \text{ GB/sec}

\text{FLOP/Byte} = 1.83 \times \frac{198}{310} \text{ GFlops (efficiency 92%)}

\text{FLOP} = 476 \quad \text{Byte} = 260

\text{310 MLUPS (Mega Lattice site Updates /sec)}

\frac{F}{B} = 6.96

Copyright © Global Scientific Information and Computing Center, Tokyo Institute of Technology
Strong/Weak Scalabilities

• For the fixed problem size, the performances are shown with increasing the number of GPUs. By introducing the overlapping technique, the performance is improved up to 30%.

• It is found that the elapsed time is shortened by increasing GPUs.

TSUBAME 2.5: 1.14 PFLOPS (3968 GPUs)
288 GFLOPS/GPU

DriVer: BMW-Audi
Lehrstuhl für Aerodynamik und Strömungsmechanik
Technische Universität München
Aerodynamics of Ping-Pong Ball
Gas-Liquid Two-Phase Flows

Two-Phase Flows

Mesh Method (Surface Capture) to solve Navier Stokes equation

- Navier-Stokes solver: Fractional Step
- Time integration: 3rd TVD Runge-Kutta
- Advection term: 5th WENO
- Diffusion term: 4th FD
- Poisson: MG-BiCGstab
- Surface tension: CSF model
- Surface capture: CLSVOF(THINC + Level-Set)

Fine meshes should be assigned around gas-liquid interfaces

Level-Set method (LSM)

The Level-Set methods (LSM) use the signed distance function to capture the interface. The interface is represented by the zero-level set (zero-contour).

\[ H(\phi) = \begin{cases} 
\frac{1}{2} & \phi > \varepsilon \\
\frac{1}{2} \left( \frac{\phi}{\varepsilon} + \frac{1}{\pi} \sin \left( \frac{\pi \phi}{\varepsilon} \right) \right) & |\phi| \leq \varepsilon \\
-\frac{1}{2} & \phi < -\varepsilon 
\end{cases} \]

Re-initialization for Level-Set function

\[ \frac{\partial \phi}{\partial \tau} = sgn(\phi) (1 - |\nabla \phi|) \]

Advantage: Curvature calculation, Interface boundary

Drawback: Volume conservation

Fig. Takehiro Himeno, et. Al., JSME, 65-635.B(1999),pp2333-2340
Sparse Matrix Solver

\[ \mathbf{A} \mathbf{x} = \mathbf{b} \quad \text{for} \quad \nabla \cdot \left( \frac{1}{\rho} \nabla p \right) = \frac{\nabla \cdot \mathbf{u}}{\Delta t} \]

Krylov sub-space methods:
- CG, BiCGStab, GMRes, ...

Pre-conditioner:
- Incomplete Cholesky,
- ILU, MG, AMG,
- Block Diagonal Jacobi

Non-zero Packing:
- CRS \rightarrow ELL, JDL

BiCGStab + (A)MG

Set \( k = 0 \) \( r_0 = p_0 = M^{-1}(\mathbf{b} - \mathbf{A}x_0) \)

for \( k = 0; k < N; k++ \)

\[ \alpha_k = \frac{(r_0, r_k)}{(r_0, M^{-1}Ap_k)} \quad q_k = r_k - \alpha_k M^{-1}Ap_k \quad \omega_k = \frac{(q_k, M^{-1}Ap_k)}{(M^{-1}Ap_k, M^{-1}Ap_k)} \]

\[ x_{k+1} = x_k + \alpha_k p_k + \omega_k q_k \]

\[ r_{k+1} = q_k - \omega_k M^{-1}Ap_k \]

if \( (r_{k+1}, r_{k+1}) < \epsilon^2 (\mathbf{b}, \mathbf{b}) \) exit;

\[ \beta_k = \frac{(r_0, r_{k+1})}{\omega_k (r_0, M^{-1}Ap_k)} \]

\[ p_{k+1} = r_{k+1} + \beta_k (p_k - \omega_k M^{-1}Ap_k) \]

loop end
MG V-Cycle

Smother : Red & Black ILU

\[ A^F : \text{Fine Matrix} \]
\[ A^C : \text{Coarse Matrix} \]
\[ R : \text{Restriction} \]
\[ P : \text{Prolongation} \]

\[ A^C = R A^F P \]

A drop on the dry floor
Comparison with Experiment

Simplified Model for a Transmission Gear

Golf Bunker Shots

Copyright © Global Scientific Information and Computing Center, Tokyo Institute of Technology
Simulation for **Granular Materials**

**DEM (Discrete Element Method)**

Contact interaction

\[ F_{ij} = -kx_{ij} - \gamma \dot{x}_{ij} \]

Normal direction

Tangential direction

Dynamic Load Balance

- **2 dimensional slice-grid method**

Many particles

1. Move \( \mathcal{X} \) boundary

2. Move \( \mathcal{Y} \) boundary

Copyright © Global Scientific Information and Computing Center, Tokyo Institute of Technology
Scalability on TSUBAME2.5

16.7 millions particles with 64 GPUs

2,018,480 particles
16,146,720 particles
129,170,880 particles

Copyright © Global Scientific Information and Computing Center, Tokyo Institute of Technology
Development of New Materials

- Mechanical Structure
- Microstructure

Low-carbon society

- Improvement of fuel efficiency by reducing the weight of transportation and mechanical structures
- Developing lightweight strengthening material by controlling microstructure

Phase-Field Model

The phase-field model is derived from non-equilibrium statistical physics and $f = 0$ represents the phase A and $f = 1$ for phase B.

Phase-field $\phi$

Phase B = Liquid
Phase A = Solid

diffusive interface with finite thickness
Al-Si: Binary Alloy

Time evolution of the phase-field $\phi$
(Allen-Cahn equation)

$$\frac{\partial \phi}{\partial t} = M_\phi \left[ \nabla \cdot \left( a^2 \nabla \phi \right) + \frac{\partial}{\partial x} \left( a \frac{\partial a}{\partial \phi_x} |\nabla \phi|^2 \right) + \frac{\partial}{\partial y} \left( a \frac{\partial a}{\partial \phi_y} |\nabla \phi|^2 \right) + \frac{\partial}{\partial z} \left( a \frac{\partial a}{\partial \phi_z} |\nabla \phi|^2 \right) - \Delta S \Delta T \frac{dp(\phi)}{d\phi} - W \frac{dq(\phi)}{d\phi} \right]$$

Time evolution of the condensation: $c$

$$\frac{\partial c}{\partial t} = \nabla \cdot \left[ D_S \phi \nabla c_S + D_L (1 - \phi) \nabla c_L \right]$$

Finite Difference Method

Phase Field: $\phi_{i,j,k}$ 19 points to solve

Condensation: $c_{i,j,k}$ 7 points to solve
Hybrid-YZ Overlapping method

Hybrid-Y Overlapping method
**Strong Scalability**

- **GPU-Only** (No overlapping)
- **Hybrid-YZ** (y,z boundary by CPU)
- **Hybrid-Y** (y boundary by CPU)

**Mesh size:**
- **512³**
- **1024³**
- **2048³**

**Performance [TFlops]**

**Number of GPUs**

**Weak Scalability:** 2.0000 PFLOPS on 4,000 TSUBAME 2.0, 330 billion cells
44.5% the peak performance
SUMMARY

■ Successful GPU Applications with structured mesh on GPU-based supercomputer TSUBAME 2.5
  • Weather Prediction
  • Aerodynamics by Lattice Boltzmann Method
  • Two-phase Flow
  • Granular material simulation
  • Dendritic Solidification by Phase-Field model

■ Performance model for multi-GPU computation works well and shows pessimistic performance for forthcoming supercomputers.

■ Studies on Framework, Algorithm for new architectures.
  • new algorithms and new numerical methods to increase arithmetic (compute) intensity