Effective Dynamic Load Balance using Space-Filling Curves for Large-scale SPH Simulations on GPU-rich Supercomputers

Dr. Satori Tsuzuki
(Japan Agency for Marine-Earth Science and Technology)

Prof. Takayuki Aoki
(Tokyo Institute of Technology)
SPH (Smoothed Particle Hydrodynamics) is a short-range interaction based particle method for simulations of incompressible flows.

In SPH, physical properties are considered to be distributed in the neighboring area in the range of an effective radius of smoothing functions $W$.

\[
\phi(x) = \int_{r \in W} \phi(r) W(x - r) \, dr
\]

**Normalization**
\[
\int W(x - r) \, dr = 1
\]

**Discretization**
\[
\phi(x) = \sum_j m_j \frac{\phi_i}{\rho_j} W(x - x_j)
\]

$m_j$ : mass  \hspace{1cm} $\rho_j$ : density
A short-range interaction based particle method for granular materials
Interactions among particles are calculated only when they contact.

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

- Interactions among only particles in the neighboring cells

**Neighbor-particle List Method**

- Interactions among only particles which exist in neighboring cells are calculated.

- **Neighbor-particle list** is often coupled with **linked-list technique** to reduce the amount of memory consumption.

- It becomes possible to reduce the interaction’s costs from $O(N^2)$ to $O(N)$.

Computational costs is **directly proportional to the number of particles**.
GPU Computing

- GPU (Graphics Processing Units)

- Multi-GPU computing

Effective methods for multi-GPU computing should be considered.
TSUBAME 2.5

**Compute Node**
(2 CPUs, 3 GPUs)

- Performance: **4.08** TFLOPS
- Memory: 58.0GB(CPU) + **18** GB(GPU)

**Rack** (30 nodes)
- Performance: **122** TFLOPS
- Memory: 2.28 TB

Performance: 224.7 TFLOPS (CPU) ※ Turbo boost
- **5.562** PFLOPS (GPU)
- Total: **5.7** PFLOPS (double precision)
- **17.1** PFLOPS (single precision)

Memory: 116 TB
Multi-GPU Computing

- A load imbalance problem among subdomains

Many particles

Few particles

GPU
Dynamic Load Balance

- 2-dimensional slice-grid method (2012)

1. Boundary shift for **vertical** direction

2. Boundary shift for **horizontal** direction
Dynamic Load Balance

- Particle counting and boundary shift on GPU
De-fragmentation of GPU memory

1. Inflows/outflows of particles

2. Re-numbering of particle data

Each Number represents the domain ID.
Verification

- The passive particles under vortex velocity field using 64 GPUs
An agitation simulation using 64 GPUs (2013)
A Golf Bunker Shot Simulation

An arrangement of golf bunker shot simulation

An example of physical conditions

<table>
<thead>
<tr>
<th>Physical Condition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proper time</td>
<td>$5.0 \times 10^{-6}$ [s]</td>
</tr>
<tr>
<td>Radius</td>
<td>0.4 [mm]</td>
</tr>
<tr>
<td>Mass</td>
<td>$5.09 \times 10^{-7}$ [kg]</td>
</tr>
<tr>
<td>Young's module</td>
<td>2.8 [GPa]</td>
</tr>
<tr>
<td>Poisson ratio</td>
<td>0.17</td>
</tr>
<tr>
<td>Friction co-efficient</td>
<td>0.3</td>
</tr>
<tr>
<td>Number of particles</td>
<td>$1.67 \times 10^7$</td>
</tr>
<tr>
<td>Time steps</td>
<td>104000</td>
</tr>
</tbody>
</table>
A golf bunker shot simulation using 16.7 millions particles with 64 GPUs (2013)
A Dam Break Simulation

Arrangement and physical conditions of incompressible flows

<table>
<thead>
<tr>
<th>Tab. 1: Conditions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$g$ [m/s$^2$]</td>
<td>9.8</td>
</tr>
<tr>
<td>$\nu$ [m$^2$/s]</td>
<td>$1.0 \times 10^{-6}$</td>
</tr>
<tr>
<td>$\rho$ [kg/m$^3$]</td>
<td>$1.0 \times 10^3$</td>
</tr>
<tr>
<td>$\nu_{max}$ [m/s]</td>
<td>12</td>
</tr>
<tr>
<td>$l_0$ [m]</td>
<td>0.0125</td>
</tr>
<tr>
<td>$C_h$</td>
<td>2.6</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>2</td>
</tr>
<tr>
<td>$C_T$</td>
<td>0.5</td>
</tr>
<tr>
<td>$C_M$ [m/s]</td>
<td>2/15</td>
</tr>
</tbody>
</table>
A dam break simulation using 72 M particles with 80 GPUs (2014)
Strong Scalability

Computational conditions

- Number of particles: $2 \times 10^6$, $1.6 \times 10^7$, $1.29 \times 10^8$ particles
- Decomposition method: Slice-grid method with dynamic load balance
- Time integration: 2nd order Runge-Kutta method
- Problem: Agitation simulation

Allocate single GPU per each subdomains.

Define performance $P$ as follows:

$$P = \left( \text{Computational time/steps} \right)^{-1} \times \text{Number of particles}$$
Strong Scalability

![Graph showing strong scalability with data points for 20,184,800, 16,146,720, and 12,917,088 particles. The x-axis represents the number of GPUs, and the y-axis represents performance in cells/s. The graph also includes an idle line.](image)
Space-Filling Curve

- A space-filling curve enables us to fill a multi-dimensional space with a continuous curve.
- A computational domain becomes recursively divided by a tree, and leafs are connected by using a space-filling curve so that each bundle of leafs has same number of particles.
A space-filling curve enables us to fill a multi-dimensional space with a continuous curve. A computational domain becomes recursively divided by a tree, and leafs are connected by using a space-filling curve so that each bundle of leafs has same number of particles.
Hilbert Curve

- A computational domain becomes recursively divided into 2x2 sub-leaves by a quad-tree.
- The Hilbert curve shows good connectivity among sub-leaves.
Morton Curve

- A computational domain becomes recursively divided into 2x2 sub-leaves by a quad-tree.
- The Morton curve (Z-curve) often causes large jumps between sequential leaves.
Peano Curve

- A computational domain becomes recursively divided into $3 \times 3$ sub-leaves by a nona-tree.
- The Peano curve shows high locality because it divides the leaf into $3 \times 3$ sub-leaves.
Weak Scalability

Computational conditions

<table>
<thead>
<tr>
<th>Conditions</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>particles / (GPU)</td>
<td>441.535</td>
</tr>
<tr>
<td>total number of GPUs</td>
<td>256</td>
</tr>
<tr>
<td>total number of particles</td>
<td>1.7 M (million), 4 M, 111 M particles</td>
</tr>
<tr>
<td>problem size / (GPU)</td>
<td>9.0 m × 8.0 m × 10 m</td>
</tr>
<tr>
<td>Supercomputer</td>
<td>TSUBAME2.5 (Tokyo Tech, JAPAN)</td>
</tr>
<tr>
<td>CPU</td>
<td>XeonX5670, 2Socket (12Core)</td>
</tr>
<tr>
<td>GPU</td>
<td>NVIDIA Tesla K20X GPUs</td>
</tr>
</tbody>
</table>

- Allocate single GPU per each subdomains.
- Define performance $P$ as follows:

$$P = (\text{computational time/steps})^{-1} \times \text{number of particles}$$
Weak Scalability

- Slice Grid
- Ideal Case

Slice Grid (2D) 49%

$P_{\text{weak}}$ [s$^{-1}$]

Number of GPUs

1.766,143
Particles

14,152,248
Particles

111,896,543
Particles

Copyright © Satori Tsuzuki, Japan Society for the Promotion and Science (JSPS)
Weak Scalability

- Hilbert (2D) 96%
- Morton (2D) 95%
- Peano (2D) 85%
- Slice Grid (2D) 49%

Graph shows the weak scalability of different data structures with varying number of GPUs and particles.
Strong Scalability

\[ P_{\text{strong}} [s^{-1}] \]

- Ideal Case
- Slice Grid

Number of GPUs

- 4 GPUs
- 32 GPUs
- 256 GPUs

Particles:
- 1,766,143
- 14,152,248
- 111,896,543

Copyright © Satori Tsuzuki, Japan Society for the Promotion and Science (JSPS)
Strong Scalability

![Graph showing the scalability of different algorithms with varying number of GPUs.](image)
Connectivity of Sub-domains

- Maximum number of connections among neighboring subdomains.
A Tsunami Simulation Using 20 Million Particles Including 945 Rubbles with 256 GPUs (2015)
A Debris Flow Simulation Using 117 Million Particles
Including 10,368 Rubbles with 256 GPUs (2015)
Dynamic Load Balance

- Dynamic load balance with Hilbert Curves (All the 117M particles are visualized.)
Conclusion

We succeeded in realizing large-scale (DEM/SPH) simulations with using from $10^7$ to $10^8$ particles by applying effective methods of dynamic load balance among GPUs based on the slice-grid method / space-filling curves on the TSUBAME2.5 supercomputer.

- The weak and strong scalability of a test case SPH simulation are examined on the multi-GPU system. It is found that the performance keeps improving in proportion to the number of GPUs when using space-filling curves.

- A realistic large-scale tsunami simulation with 7 buildings and floating driftwoods was successfully carried out running on 256 GPUs on TSUBAME 2.5 at Tokyo-Tech.

- A debris flow simulation using 117 Million particles interacting with 10,638 floating rubbles was successfully executed running on 256 GPUs on TSUBAME 2.5 at Tokyo-Tech.