Large-scale Simulation of the Global SWEs on Hybrid CPU-GPU Platforms

Lin Gan
Computer Science & Technology
Center for Earth System Science
Tsinghua University, China
Email: l-gan11@mails.tsinghua.edu.cn
Tel: +86-15810537053

Sep 13, 2012 @ NCAR, Boulder, US
A team work over the last 5 months:

- Chao Yang (Institute of Software Chinese Academy of Sciences)
- Lin Gan, Wei Xue, Haohuan Fu, Yangtong Xu (Center for Earth System Science, Tsinghua University)

Achievements:

- Peta-scalable global SWE simulation, 809TFlops in double precision in 3750 nodes (45,000 CPU cores & 52,500 GPU cores)
- Adjustable partition between CPUs/GPUs
- Effective comm.-comp. overlap to hide comm. cost
- “Pipe-flow” scheme to arrange message-passing
- Systematic optimizations on CPU and CUDA code, 130 speedup
Global atmospheric modeling is a key component in climate simulation.

- Land model: petascale-ready
- Ice model: petascale-ready
- Ocean model: nearly petascale-ready
- Atmosphere model: bottleneck

(Courtesy: Mark Taylor et al., 2009)

Note: petascale-ready means scalable to $O(100\text{K})$ processors at target climate resolution (<10km)
Rapid development of heterogeneous supercomputers

- On the recent Top500 list (Jun 2012)
  - #5: Tianhe-1A, 4.7 Pflops, 86K CPU-cores + 100K GPU-cores
  - #10: Nebulae, 2.98 Pflops, 55K CPU-cores + 65K GPU-cores

- Coming soon
  - Titan, successor to Jaguar, ORNL
  - Blue Waters, NCSA

Successes in hybrid CPU-GPU algorithms

- N-body simulations (Hamada et al. SC’09, GB winner)
- NWP simulations (Takashi Shimokawabe et al. SC’10)
- Biofluidics simulations (Bernaschi et al. SC’11, GB final list)
- Phase-field simulations (Shimokawabe et al. SC’11, GB winner)

Efficient hybrid algorithms in global atmospheric modeling
Contents

1. Background
2. Mesh & Equations
3. Algorithm & Implementations
4. Large-scale results
5. Conclusions
Mesh & Equations

✧ Equations: Shallow Water Equations (SWEs)

- SWEs exhibit most basic horizontal dynamics of the global atms

\[
\begin{align*}
\frac{\partial h}{\partial t} + \nabla \cdot (hv) &= 0, \\
\frac{\partial hv}{\partial t} + \nabla \cdot (hv \otimes v) + gh\nabla (h + b) + fh\mathbf{k} \times v &= 0
\end{align*}
\]
Mesh & Equations

🔹 Cubed-sphere mesh
  ▪ Mapping inscribed cube to the surface of the earth

🔹 Computational domain
  ▪ Six patches covered with rectangular meshes
Mesh & Equations

✧ 13-point stencil

✧ Interp. Across patches
  - 1-d linear interpolation

✧ Discretize using cell-centered finite volume scheme

✧ Intergrade using second order TVD Runge-Kutta method
Algorithms & Implementations

In this part:
❖ Tianhe-1A supercomputers
❖ Hybrid algorithm
❖ Pipe-flow scheme
❖ Systematic optimization
Algorithms & Implementations

.getJSONObject("The Tianhe-1A")

- TH-net: fast proprietary network
- 7168 computing nodes, 4.7PFlops Rpeak (#5 in Top500, 2012)
- In each computing node
  - Two 6-core Intel X5670 CPUs: 12 cores, 140Mflops
  - One NVIDIA M2050 GPU: 14 cores (448 CUDA cores), 515Mflops

lin.gan27@gmail.com
Domain decomposition of the cubed-sphere

- 6 MPI groups for 6 patches
- Divide each patch into N*N sub-blocks (here 4*4)

Mesh points (filled) and halos (empty) for a sub-block
Algorithms & Implementations

CPU-only algorithm

For each stencil cycle
① Update halos
② Prepare buffer copy
③ Interpolate ghost cells if necessary
④ Calculate stencils
CPU-only algorithm: work flow

Using OpenMP to exploit the 12 CPU cores in each node of the TH-1A
Hybrid CPU-GPU algorithm

- Divide each sub-block into
  - Outer part: 2 layers of halos for neighbors → CPU computing
  - Inner part: without halo exchanging → GPU computing

(Courtesy: J. Michalakes et al., SC11)
Hybrid CPU-GPU algorithm

- Divide each sub-block into
  - Outer part: 2 layers of halos for neighbors → CPU computing
  - Inner part: without halo exchanging → GPU computing

For each stencil cycle

**GPU side:**
1. Calculate stencils in inner part

**CPU side:**
1. Update halos
2. Prepare buffer copy
3. Interpolate ghost cells if necessary
4. Calculate stencils in outer part

**BARRIER:**
5. Exchange data between CPU/GPU
Hybrid CPU-GPU algorithm: work flow

- MPI: update halos
- CPU: copy, interp.
- GPU: stencil comput. for inner part

One stencil cycle:
- halo-top
- halo-bottom
- halo-left
- halo-right

Steps:
1. Update halos
2. Copy
3. Interp.
4. Stencil computation
5. G2C
6. C2G
7. Barrier

lin.gan27@gmail.com
Adjustable partition CPU-GPU algorithm

- Divide each sub-block into:
  - Outer part: n layers of points → CPU computing
  - Inner part: without halo exchanging → GPU computing
Hybrid CPU-GPU algorithm: improvement

- Communication (halo updating) degrades CPU flops efficiency
  - Strategy for communication-computation overlap
- Imbalanced message-passing on the cubed-sphere
  - e.g. Patch 0,1,2,3 might send top halos to Patch 5 at the same time
Improved hybrid CPU-GPU algorithm

“pipe-flow” scheme for message-passing on cubed-sphere

Four steps to arrange conflict-free message-passing on cubed-sphere.

The arrows indicate directions of data entering or exiting patches as a pipe flow.
Improved hybrid CPU-GPU algorithm

“pipe-flow” scheme for message-passing on cubed-sphere

Directions of the “pipe-flow” inside a particular patch of the cubed-sphere.

The square shaded regions represent the inner parts of sub-blocks.

The narrow shade regions represent the halos to be sent.

“pipe-flow” pattern for the 4*4 sub-blocks in a patch
Improved hybrid CPU-GPU algorithm: work flow

Note: halo1/2/3/4 — the 4 steps of the “pipe-flow” communication scheme
adjustable partition between CPU and GPU
CPU optimizations

Mesh size: 1156*1156

1. Baseline
2. OpenMP `nowait` clause
3. Tuning number of threads to twice the CPU cores
4. OpenMP task construct
5. Tuning the best task granularity
GPU optimizations

Mesh size: 1024*1024

1. Baseline
2. Computing instead of reading pre-computed auxiliary vectors (e.g., tensors)
3. Increasing L1 cache from 16KB to 48KB, without using the shared memory (SM)
4. 48KB L1 cache + 16KB SM
5. 16KB L1 cache + 48KB SM

130 speedup over CPU kernel
Large-scale tests

✧ Validation of the model
  - Isolated mountain (Williamson test set, JCP 1992)
    Day 15, 10,240*10,240*6 mesh (1km res), 1536 nodes of TH1A

lin.gan27@gmail.com
Large-scale tests

✧ Validation of the model (cont’d)
  - Real topography of the Earth, zonal flow
    Day 15, 256*256*6 mesh (40km res)

lin.gan27@gmail.com
Large-scale tests

✧ Validation of the model (cont’d 2)

- Real topography of the Earth, zonal flow
  Day 15, 10,240*10,240*6 mesh (1km res)
### Large-scale tests

#### Weak-scaling tests: configurations

<table>
<thead>
<tr>
<th>Number of nodes</th>
<th>Mesh size</th>
<th>Peak of (CPU, GPU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$6 = 6 \times 1 \times 1$</td>
<td>$6 \times 1024 \times 1024$</td>
<td>(0.8, 3.1) Tflops</td>
</tr>
<tr>
<td>$24 = 6 \times 2 \times 2$</td>
<td>$6 \times 2048 \times 2048$</td>
<td>(3.3, 12) Tflops</td>
</tr>
<tr>
<td>$96 = 6 \times 4 \times 4$</td>
<td>$6 \times 4096 \times 4096$</td>
<td>(14, 49) Tflops</td>
</tr>
<tr>
<td>$384 = 6 \times 8 \times 8$</td>
<td>$6 \times 8192 \times 8192$</td>
<td>(54, 198) Tflops</td>
</tr>
<tr>
<td>$864 = 6 \times 12 \times 12$</td>
<td>$6 \times 12288 \times 12288$</td>
<td>(121, 445) Tflops</td>
</tr>
<tr>
<td>$1536 = 6 \times 16 \times 16$</td>
<td>$6 \times 16384 \times 16384$</td>
<td>(216, 791) Tflops</td>
</tr>
<tr>
<td>$2400 = 6 \times 20 \times 20$</td>
<td>$6 \times 20480 \times 20480$</td>
<td>(337, 1236) Tflops</td>
</tr>
<tr>
<td>$2904 = 6 \times 22 \times 22$</td>
<td>$6 \times 22528 \times 22528$</td>
<td>(408, 1496) Tflops</td>
</tr>
<tr>
<td>$3750 = 6 \times 25 \times 25$</td>
<td>$6 \times 25600 \times 25600$</td>
<td>(527, 1931) Tflops</td>
</tr>
</tbody>
</table>
Large-scale tests

Weak-scaling results

- Largest run: 3750 nodes, 25,600*25,600*6 mesh (11.8 B unknowns)

<table>
<thead>
<tr>
<th></th>
<th>CPU-only (1-core)</th>
<th>CPU-only (12-core)</th>
<th>CPU-GPU</th>
<th>CPU-GPU tuned</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregate performance (Tflops)</td>
<td>11.5 Tflops</td>
<td>126 Tflops</td>
<td>658 Tflops</td>
<td>809 Tflops</td>
</tr>
</tbody>
</table>

32.8% of peak!
Strong-scaling results (N=25,200, 100 time steps)

<table>
<thead>
<tr>
<th>Number of nodes</th>
<th>384</th>
<th>1536</th>
<th>2400</th>
<th>3750</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time (s)</td>
<td>56.9</td>
<td>14.4</td>
<td>9.4</td>
<td>5.9</td>
</tr>
<tr>
<td>Efficiency</td>
<td>1.00</td>
<td>0.99</td>
<td>0.97</td>
<td>0.98</td>
</tr>
<tr>
<td>Agg. Tflops</td>
<td>84.5</td>
<td>335.1</td>
<td>513.4</td>
<td>809.6</td>
</tr>
</tbody>
</table>

G2C/C2G as the non-overlapping part is very small

Successful comm.-comp. overlap
Conclusions

✧ A scalable approach for global SWEs on CPU-GPU clusters
  ▪ Adjustable partition between CPUs/GPUs
  ▪ Effective comm.-comp. overlap to hide comm. cost
  ▪ “Pipe-flow” scheme to arrange message-passing
  ▪ Systematic optimizations on CPU and CUDA code

✧ Ideal weak/strong scaling on the Tianhe-1A
  ▪ Up to 3750 nodes (45,000 CPU cores + 52,500 GPU cores)
  ▪ Sustaining 809 Tflops (32.8% of peak) on 3750 nodes

✧ SWEs on FPGA
  ▪ Using Maxeler Xlinx FPGA clusters
  ▪ Estimate a further improvement
Thank You!

Presenter: Lin Gan

Computer Science & Technology
Center for Earth System Science
Tsinghua University, China
Email: l-gan11@mails.tsinghua.edu.cn
Tel: +86-15810537953