Multicore/manycore with a scientific computing culture: experience feedback with special focus on performance modeling and redesign of computational methods

Florian De Vuyst

CMLA, ENS Cachan Université Paris-Saclay et CNRS

37è Forum ORAP – 17 mars 2016, CNRS Michel-Ange, Paris
Acknowledgements

The team :
- Marie Béchereau (PhD candidate, CMLA)
- Thibault Gasc (PhD candidate, MDLS, CEA, CMLA)
- Mathieu Peybernes (CEA Saclay, DEN)
- Raphael Poncet (CGG)
- Renaud Motte (CEA DAM DIF)

Acknowledgements :
- Thomas Guillet, Philippe Thierry (INTEL)
- NVIDIA (CUDA Research Center, 2013)
- MDLS for hardware support
1. GPU computing experience
Beginning of the story ...
Real-time interaction of 2D flow dynamics on boards/screens (2012)
Beginning of the story ...
Real-time interaction of 2D flow dynamics on boards/screens (2012)
GPU computing experience

- NVIDIA CUDA Language, TESLA boards
- Substantial speedup observed
- Nice binding between computation and visualization

- Classical 2D FV CFD Cartesian code & LBM codes implemented

- Performance issue: sometimes hard to choose the adequate data structures
- Global performance: (most of the time) hard to understand
2. Performance modeling on (multicore) CPU
Performance modeling: objectives

- Derive a model which is able to predict the performance of a given solver on a given CPU-based machine.
Elements of a performance model

- Computing architecture
- Algorithm / numerical scheme
- Performance model
- Predicted performance
Reference node-based performance model: Roofline

[S. Williams, A. Waterman & D. Patterson, Roofline: an insightful visual performance model for multicore architectures, Communications ACM 55(6): 121-130 (2012)]
1. GPU computing experience
2. Performance modeling on (multicore) CPU
3. Application: Lagrangian remap Hydrodynamics solver

Execution Cache Memory (ECM) model
[Hager & Treibig 2010]

Metrics: time [cycle / cache line update]
- Execution times at (ALU+L1 cache) level
- Transfer speeds between Lx-L(x+1) cache/memory

[G. Hager et al., Concurrency and Computation: Practice and Experience, DOI: 10.1002/cpe.3180 (2013).]
1. GPU computing experience

2. Performance modeling on (multicore) CPU

3. Application: Lagrangian remap Hydrodynamics solver

ECM: from one-core to multi-core CPU
3. Application: Lagrangian remap Hydrodynamics solver
1. GPU computing experience
2. Performance modeling on (multicore) CPU
3. Application: Lagrangian remap Hydrodynamics solver

Lagrangian remap algorithm
1. GPU computing experience
2. Performance modeling on (multicore) CPU
3. Application: Lagrangian remap Hydrodynamics solver

Dataflow diagram

**Figure:** LEFT: Lagrangian step & RIGHT: remapping step dataflow diagrams (input data, kernels, output data)
Results

- Roofline is able to predict performance of compute-bound kernels ($O(5\%)$ errors)
- ECM is necessary for good performance prediction of memory-bound kernels ($O(5\%)$ errors)
Redesign of Lagrangian remapping

Why is it necessary to redesign?

- Staggered velocity variables involve more communications (memory-bound)
- Alternating direction strategies multiply communications too
- The remapping process is not SIMD
Focus: remapping process

Figure: Geometric remapping: bad for SIMD

Figure: Geometry-free reformulation by balance of advection fluxes: SIMD-ready ⇒ Lagrange-flux schemes
Comparison of results between legacy Lagrange-remap and Lagrange-flux scheme
Multimaterial extension of Lagrange-flux schemes
Dataflow diagram for the Lagrange-flux scheme

Rem : easy parallelization & vector by #pragma directives.
### Performance prediction vs measurements with ECM

<table>
<thead>
<tr>
<th>Kernel name</th>
<th>type</th>
<th>prediction</th>
<th>measure</th>
<th>error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Lagrange-Remap</td>
<td>MB</td>
<td>10697 cy/CL</td>
<td>10092 cy/CL</td>
<td>6%</td>
</tr>
<tr>
<td>Prediction Lagrange-Flux</td>
<td>CB</td>
<td>5712 cy/CL</td>
<td>5548 cy/CL</td>
<td>2.8%</td>
</tr>
<tr>
<td>Correction Lagrange-Flux</td>
<td>CB</td>
<td>5712 cy/CL</td>
<td>5690 cy/CL</td>
<td>0.3%</td>
</tr>
</tbody>
</table>

**Figure:** Performance metrics: nb. of cycles (cy) per cache line (CL)
Comparison of vector + multicore scalability performance

<table>
<thead>
<tr>
<th>Scheme</th>
<th>1 core</th>
<th>1 core AVX</th>
<th>16 cores AVX</th>
<th>scalability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lagrange-Flux</td>
<td>1.9</td>
<td>3.9</td>
<td>52.0</td>
<td>27.1</td>
</tr>
<tr>
<td>Lagrange-Remap</td>
<td>2.4</td>
<td>3.7</td>
<td>36.5</td>
<td>14.6</td>
</tr>
</tbody>
</table>

Figure: Performance comparison in millions of cell updates (MCUPs), CPUs INTEL SandyBridge $2 \times 8$ cores. Performed on a fine mesh.
1. GPU computing experience
2. Performance modeling on (multicore) CPU
3. Application: Lagrangian remap Hydrodynamics solver

Research perspective

- Extend the performance modeling methodology to GPU/manycore-based
- (NB Some literature already exists on this subject)
Concluding remarks

- Nice speedups with GPU but sometimes unclear performance issues
- Performance modeling is a powerful tool to understand global performance
- Performance analysis is insightful for bottleneck analysis
- Some parts of the solver may be redesigned for achieving better arithmetic intensity, leading to new computational approaches
- Future work: accurate performance models for accelerators/manycore/GPU proc.?

