Challenges in fluid flow simulations using Exa-scale computing

Mahendra Verma
IIT Kanpur

http://turbulencehub.org

mkv@iitk.ac.in
Hardware
A Growth-Factor of a Billion in Performance in a Career

<table>
<thead>
<tr>
<th>Year</th>
<th>Performance Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1950</td>
<td>1 KFlop/s</td>
</tr>
<tr>
<td>1960</td>
<td>1 MFlop/s</td>
</tr>
<tr>
<td>1970</td>
<td>1 GFlop/s</td>
</tr>
<tr>
<td>1980</td>
<td>1 TFlop/s</td>
</tr>
<tr>
<td>1990</td>
<td>1 PFlop/s</td>
</tr>
<tr>
<td>2000</td>
<td>100 PF</td>
</tr>
<tr>
<td>2010</td>
<td>131 PF</td>
</tr>
</tbody>
</table>

1941: 1 (Floating Point operations / second, Flop/s)
1945: 100
1949: 1,000 (1 KiloFlop/s, KFlop/s)
1951: 10,000
1961: 100,000
1964: 1,000,000 (1 MegaFlop/s, MFlop/s)
1968: 10,000,000
1975: 100,000,000
1987: 1,000,000,000 (1 GigaFlop/s, GFlop/s)
1992: 10,000,000,000
1993: 100,000,000,000
1997: 1,000,000,000,000 (1 TeraFlop/s, TFlop/s)
2000: 10,000,000,000,000
2005: 131,000,000,000,000 (131 TFlop/s)

From Karniakadis’s course slides
### AMD EPYC™ 7551

**Specifications**

- **# of CPU Cores:** 32
- **Max Boost Clock:** 3GHz
- **Socket Count:** 1P/2P
- **All Core Boost Speed:** 2.55GHz
- **PCI Express Version:** x128
- **Base Clock:** 2GHz
- **Total L3 Cache:** 64MB
- **Default TDP / TDP:** 180W

**System Memory**

- **System Memory Specification:** 2666MHz
- **Memory Channels:** 8
- **Mem BW (2S Theo):** 341 GB/s

---


**NODE:** 2 proc/node; Focus on a node

**Flop rating for 2 procs:** $2 \times 32 \times 24 = 1536$ GF

**Wants data ~ 8 TB/sec.**

**Cache, RAM, HD**
Data transfer

FLOPS free, data transfer expensive (Saday)

Memory BW = 341 GB/s

SSD: transfer rate = 6 Gbit/s

peak IB Switch speed/port = 200 Gb/s
software challenges
For beginners

- Abundance (MPI, OpenMP, CUDA, ML)
- Leads to confusion and non-start..
- Structured programming
- Pressure to do the science..
- Some times CS tools are too complex to be practical.
For advanced users

- Optimised use of hardware.
- Structured and modular, usable code with documentation.
- Keeping up with upgrades and abundance (MPI3, ML, C++11, Vector processors, GPU, XeonPhi, Rasberry Pi).
- Optimization
- Interactions with users + programers
Now CFD

(Computational fluid dynamics)
Applications

• Weather prediction and climate modelling
• Aeroplane and cars (transport)
• defence / offences
• Turbines, dams, water management
• Astrophysical flows
• Theoretical understanding
Field reversal

with Mani Chandra
Polarity reversals after random time intervals (tens of millions of years to 50K years).

Last reversal took place around 780,000 years ago.

Glatzmaier & Roberts
Nature, 1995
Nek5000 (Spectral-element) simulation

Chandra & Verma, PRE 2011, PRL 2013

spectral-element code Nek5000

$(1,1) \rightarrow (2,2) \rightarrow (1,1)$
Methods

- Finite difference
- Finite volume
- Finite element
- Spectral
- Spectral element
Spectral method
Example: Fluid solver
\[ \partial_t u + (u \cdot \nabla) u = -\nabla p + \nu \nabla^2 u + F \]

\[ \nabla \cdot u = 0 \]

Incompressibility

Reynolds number = \( \frac{UL}{\nu} \)
Procedure
\[ f(x) = \sum_{k_x, k_z} \hat{f}(k_x) \exp[i(k_x x)] \]

\[ \frac{df(x)}{dx} = \sum_{k_x, k_z} [ik_x \hat{f}(k_x)] \exp[i(k_x x)] \]
Set of ODEs

\[
\frac{du_i(k)}{dt} = -jk_m \underline{u_m(r)u_i(r)} - jk_i p(k) - \nu k^2 u_i(k)
\]

Time advance (e.g., Euler’s scheme)

\[
u_i(k,t + dt) = u_i(k) + dt \times \text{RHS}_i(k,t)
\]

Stiff equation for small viscosity \( \nu \) (use exponential trick)
Nonlinear terms computation:

(pseudo-spectral)

Fourier transforms take around 80% of total time.
Tarang = wave (Sanskrit)

Spectral code (Orszag)

One code to do many turbulence & instabilities problems

**VERY HIGH RESOLUTION (6144³)**

Cores: 196692 of Shaheen II of KAUST

Opensource, download from [http://turbulencehub.org](http://turbulencehub.org)

Chatterjee et al., JPDC 2018
Fluid
MHD, Dynamo
Scalar
Rayleigh-Bénard convection
Stratified flows
Rayleigh-Taylor flow
Liquid metal flows
Rotating flow
Rotating convection

Instabilities
Chaos
Turbulence

No-slip BC
Cylinder
sphere
Toroid
(in progress)

Periodic BC
Free-slip BC
Rich libraries to compute
Spectrum
Fluxes
Shell-to-shell transfer
Structure functions

New things
Fourier modes
Real space probes
Ring-spectrum
Ring-to-ring transfer

Tested up to $6144^3$ grids
Object-oriented design
Basis functions (FFF, SFF, SSF, SSS, ChFF)

Basis-independent universal function (function overloading)

e.g., compute_nlin \((u \cdot \nabla)u, (b \cdot \nabla)u, (b \cdot \nabla)b, (u \cdot \nabla)T\).

General PDE solver

We can use these general functions to simulate MHD, convection etc.
Generated by Doxygen
Parallelization
Spectral Transform (FFT, SFT, Chebyshev)

Multiplication in real space

Input/Output
HDF5 lib
FFT Parallelization

\[ f(x, y, z) = \sum_{k_x} \sum_{k_y} \sum_{k_z} \hat{f}(k_x, k_y, k_z) \exp[i(k_x x + k_y y + k_z z)] \]
Slab decomposition

Data divided among 4 procs
**Transpose-free FFT**

**MPI vector, conconsecutive data transfer**

<table>
<thead>
<tr>
<th>p0</th>
<th>p0</th>
<th>p1</th>
<th>p1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>17</td>
<td></td>
</tr>
</tbody>
</table>

12-15% faster compared to FFTW
Pencil decomposition

(a) Real

(b) Intermediate

(c) Fourier

(d) (e) (f)

MPI_COMM_COL

MPI_COMM_ROW

N_x

N_y

N_z
FFT scaling

On Shaheen 2 at KAUST
with Anando Chatterjee, Abhishek Kumar, Ravi Samtaney, Bilel Hadri, Rooh Khurram

Cray XC40
ranked 9th in top500

Chatterjee et al., JPDC 2018
Tarang scaling

On Shaheen at KAUST
• Weak scaling: When we increase the size of the problem, as well as number of procs, then should get the same scaling.
Average flop rating/core (~1.5 %)

Compare with BlueGene/P (~8 %)

Overlap Communication & Computation ??

GPUs ??

Xeon Phi ??
To Petascale & then Exascale
Finite difference code

General code: Easy porting to GPU, MiC

Collaborators:
Roshan Samuel
Fahad Anwer (AMU)
Ravi Samtaney (KAUST)
Summary

★ Code development
★ Module development
★ Optimization
★ Porting to large number of processors
★ GPU Porting
★ Testing
Acknowledgements

**Students:**
- Anando Chatterjee
- Abhishek Kumar
- Roshan Samuel
- Sandeep Reddy
- Mani Chandra
- Sumit Kumar & Vijay

**Faculty:**
- Ravi Samtaney
- Fahad Anwer

**Ported to:**
- PARAM, CDAC
- Shaheen, KAUST
- HPC system IITK

**Funding**
- Dept of Science and Tech., India
- Dept of Atomic Energy, India
- KAUST (computer time)
Thank you!