A Deployment of HPC Algorithm into Pre/Post-Processing for Industrial CFD on K-computer

Keiji Onishi[1], Niclas Jansson[2,1], Rahul Bale[1], Wei-Hsiang Wang[1], Chung-Gang Li[3,1] and Makoto Tsubokura[3,1]

1. RIKEN, Advanced Institute for Computational Science, Kobe, Hyogo 650-0047, Japan, E-mail: keiji.onishi@riken.jp
2. KTH Royal Institute of Technology, School of Computer Science and Communication, SE-100 44, Stockholm, Sweden
3. Kobe University, Graduate School of System Informatics, Kobe, Hyogo 657-8501, Japan

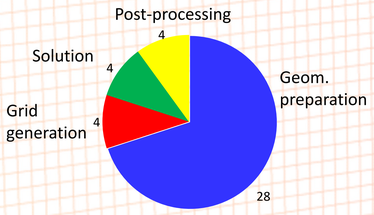
1) Background and Motivation

Pre-/post- processing is still a major problem in industrial CFD analysis.

- Pre-processing

With the rapid development of computers, physical solvers are getting faster, while pre- still remains slow because it is mainly a serial process and it is less likely of benefit from HPC technology.

For example, right figure shows an example of breakdown of turn-around time in automobile aerodynamics, where **pre- accounts for 80%** of total cost[1]. In the field of aeronautics, because pre- is required at every iteration in the design optimization cycle, processing speed up is hindered due to this bottleneck no matter how fast a solver has been made[2].



Typical turn-around time on automotive aerodynamics CFD (in hours).

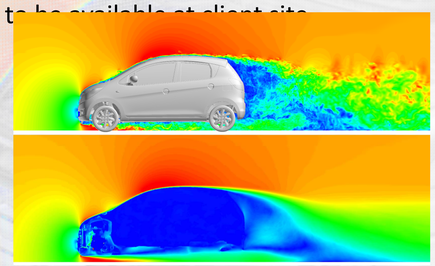
- Post-processing

Speed-up of visualization is insufficient compared to the rapidly increasing amount of data. For example, in typical industrial CFD analysis, it is reported that the total amount of **CFD data is increasing by a factor of 4 year by year**[3]. Now it is approaching Peta-Byte order.

Meanwhile, the effective disk access speed of commodity-based products is about 50 to 100 MB/s. Post will still be a bottleneck until the high-speed parallel file, visualization and, perhaps, network systems become cheap enough to be available at client site.

- Objective of this research

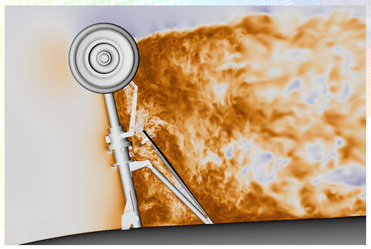
We develop a method that enables pre-/post- parallelization using HPC techniques to harness the power of modern supercomputers for the CFD

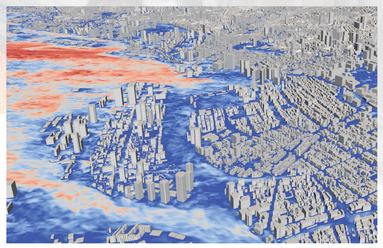


Example of automotive aerodynamics sim-ulation (top: instantaneous velocity magnitude on center plane, bottom: time averaged).

	This Method	Commercial Code A
Pre processing time	10 minutes	35 hours
Physical solver conditions	Transient (290K steps), 91M cells	RANS (5K steps), 60M elements
Calculation time of physical solver	15.1 hours *1 (54.4 hours *3)	8 hours *2
Error of ΔCd	8%	12%

General comparison with other methods
*1 K-computer, 558nodes (8core/node, SPARC64TM VIIIfx/2.0GHz)
*2 Recently designed Intel Xeon based Cluster, 512 cores
*3 SGI ICE X, 384 cores (Intel Xeon E5-2670v3/2.3GHz turbo boost)





Examples of industrial CFD (left: nose landing gear of airplane, right: wind environment of Tokyo-bay).

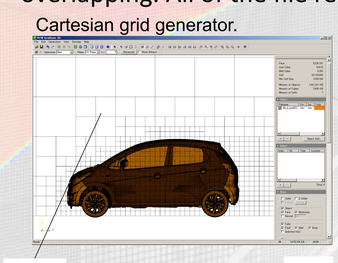
2) Pre-processing Method

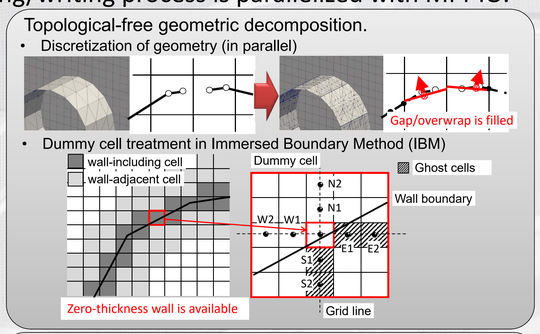
- **Topological-free geometric decomposition**

The CAD geometry is discretized into cell-oriented values based on a hierarchical Cartesian grid (BCM[4]). An arbitrary boundary representation is used with a dummy-cell technique based on immersed boundary method (IBM[5,6]). It enables physical solver to handle 'dirty' CAD without necessitating any clean-up. The physical solver should also be carefully designed to achieve reasonable accuracy based on IBM[7,8] or other methods.

- **Hybrid parallelization**

The process is parallelized with MPI+OpenMP. The data exchange between halo-region is conducted with multi-thread calculation and communication overlapping. All of the file reading/writing process is parallelized with MPI-IO.



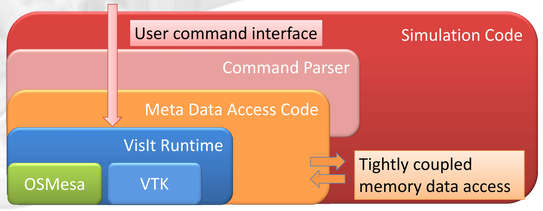


Schematic view of computational grid, and decomposed domain color (partitioned by Z-order, ParMetis, etc.).

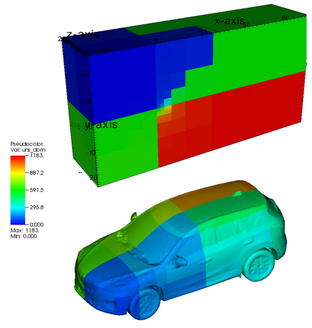
3) Post-processing Method

- **In-situ visualization using libsim**

The libsim (VisIt) library was used for in-situ visualization which requires OSMesa (OpenGL on CPU) and VTK library. The data access is tightly coupled with physical solver which has meta data access pipeline providing direct or copy access to the data on memory. The meta data is decomposed based on the BCM grid decomposition. The compilation is done on K-computer with parallel option. It runs under the batch or Interactive render mode.



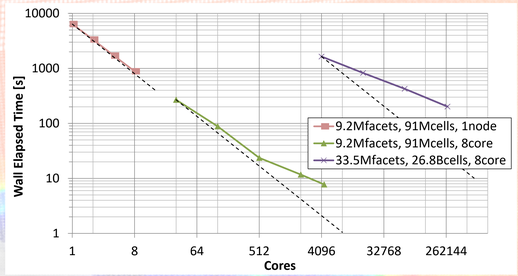
Schematic view of libsim, including command interface for user and meta-data access code.



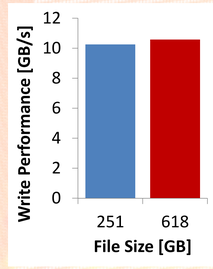
Example of decomposed domain for parallel visualization

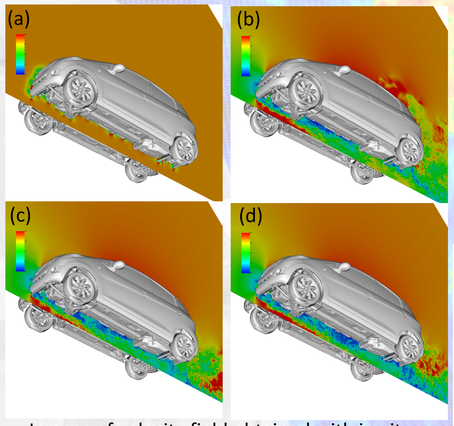
4) Results and Discussion

Pre-processing performance.



I/O performance.



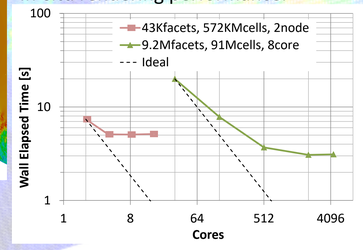


Images of velocity field obtained with in-situ visualization(a: t=0.0[s], b: t=0.2[s], c: t=0.4[s], d: t=0.6[s]).

In-situ visualization results in approx. 30% increase of entire calculation time:
- No. of flow timesteps = 290,000
- Entire calculation time = 75,743 (sec)
- No. of in-situ frames = 2901
- In-situ calculation time = 21,782 (sec)

File size of each image is just ≈200KB:
- Comparing to 2.1GB of solution data per timestep

In-situ rendering performance.



5) Conclusion

- Pre-processing, which typically requires several days of human labor, has been **reduced to the order of minutes**.
- For a data size of ~2GB/frame, the post-processing time has been **reduced to the order of several seconds**. This results in an approx. 30% increase in computational time for a vehicle aerodynamics case.

We believe these are indispensable technologies towards the EXA FLOPS CFD.

Acknowledgements

This research was supported by MEXT as "Priority Issue on Post-K computer" (Development of innovative design and production processes) and used computational resources of the K computer provided by the RIKEN Advanced Institute for Computational Science.

References

[1] Furukawa, N., Shiozawa, H., et al., "The Application of CFD Aerodynamic Computation Methods to New Vehicle Development Based on a Model that Reproduces an Engine Bay and Floor," Proc. of JSAE Annual Congress, 20045513 (2004).
[2] Todd Michal, "CSE 2009: Preprocessing for Industrial CFD: More Important Than You Might Think", SIAM news, June 15 2009, http://www.siam.org/news/news.php?id=1598, (cited: 1 Apr. 2017)
[3] Naoyuki Sogo, "Introduction of high-performance computing consulting service", VINAS Users Conference 2016, http://www.vinas.com/en/ugm2016/program.html, (cited: 1 Apr. 2017)
[4] Nakahashi, K., "Building-Cube Method for Flow Problems with Broadband Characteristic Length," *Computational Fluid Dynamics 2002*, Berlin, Heidelberg: Springer, pp. 77-81 (2003).
[5] Peskin, C. S., "Flow Patterns around the Heart Valves," *Journal of Computational Physics* **10**, 252-271 (1972).
[6] Ghias, R., Mittal, R., Dong, H., "A sharp interface immersed boundary method for compressible viscous flows," *Journal of Computational Physics,* **225**, 528-553 (2007).
[7] Onishi, K., Obayashi, S., et al., "Use of the Immersed Boundary Method within the Building Cube Method and its Application to Real Vehicle CAD Data", *AIAA paper*, 2013-2713 (2013).
[8] Onishi, K., Tsubokura, M., et al., "Vehicle Aerodynamics Simulation for the Next Generation on the K Computer: Part 2 Use of Dirty CAD Data with Modified Cartesian Grid Approach", *SAE Int. J. Passeng. Cars - Mech. Syst.* **7(2)**, 2014-01–0580 (2014).