11th World Congress on Computational Mechanics (WCCM XI)
5th European Conference on Computational Mechanics (ECCM V)
6th European Conference on Computational Fluid Dynamics (ECFD VI)
July 20–25, 2014, Barcelona, Spain

BIT-REPRESENTATION OF BOUNDARY CONDITIONS FOR HIGH PERFORMANCE INCOMPRESSIBLE THERMAL FLOW SIMULATION

Kenji Ono[1, 2]

[1] RIKEN, Advanced Institute of Computational Science, 650-0047, 7-1-26, Minatojima-minami-machi, Chuo-ku, Kobe, Japan, keno@riken.jp and http://labs.aics.riken.jp/ono.html
[2] Kobe University, Graduate School of System Infomatics

Key words: *Practical Application, Computing Method, Cartesian Mesh.*

Nowadays industrial applications demand to be executed on supercomputers because a size of calculation becomes larger and larger. Those applications are strongly required to handle various boundary conditions as well as to achieve high performance. However, calculation of boundary conditions is generally complicated, and it is not easy for compilers to optimize statements in a do-loop due to if-branch, LUT and indirect memory access. In this paper, the author presents an efficient implementation of boundary conditions using bit-repreesntation technique that enables higher performance with low memory requirement and demonstrates its benefits. The governing equations are the incompressible energy transport and Navier-Stokes equations. The equations are described in conservative form and are discretized by finite-volume method on Cartesian mesh. Eq.(1) shows the energy transport equation.

$$\rho C \left[\frac{\partial \theta}{\partial t} + \frac{\partial}{\partial x_i} (u_i \theta) \right] = -\frac{\partial q_i}{\partial x_i} + Q, \quad q_i = -\lambda \frac{\partial \theta}{\partial x_i} \tag{1}$$

where, ρ, C, θ, u_i, q_i, λ, Q are mass density, specific heat, temperature, velocity, heat flux, heat conductivity and heat source, respectively. The boundary heat flux of the diffusive part in eq.(1) will be expressed by $q_{i,BC} = (q_{i,ISO} \,|\, q_{i,T} \,|\, q_{i,C} \,|\, q_{i,D} \,|\, q_{i,A})$;

$$\left.\begin{array}{ll} Adiabatic & q_{i,A} = 0 \\ Thermal\,Conductivity & q_{i,C} = -\lambda\, \partial\theta/\partial x_i \\ Thermal\,Transmission & q_{i,T} = -H(\theta_s - \theta_\infty) \\ Isothermal & q_{i,ISO} \\ Direct\,Heat\,Flux & q_{i,D} \end{array}\right\} \tag{2}$$

Introducing the discrete heaviside function $\psi_{i,BC}(x)$ =0 (Boundary Condition), 1 (Non-BC) at the each cell face to be imposed a boundary condition, heat flux in eq.(1) will be replaced by $q_i = \psi_{BC}\, q_i + (1 - \psi_{BC}) q_{i,BC}$. Finally, semi-discrete form of eq.(1) will be

$$\rho C \frac{\theta^{n+1} - \theta^n}{\Delta t} = -\frac{\rho C}{h} \sum_{j=face}^{1\sim6} (u\,\theta)_j\, n_j + \frac{1}{h^2} \left[\sum_{j=face}^{1\sim6} (\psi_{BC}\lambda)_j\, \theta_J - \theta_P \sum_{j=face}^{1\sim6} (\psi_{BC}\lambda)_j \right] - \frac{1}{h} \sum_{j=face}^{1\sim6} \{(1-\psi_{BC})\, q_{BC}\}_j\, n_j + Q \tag{3}$$

where, h, j, n_j, J, P represent mesh width, position of cell face, normal vector at cell face, temperature of neighbor cell corresponds to j direction and present cell, respectively. Eq.(3) can be solved by both explicit and implicit schemes. The heat source term Q and boundary conditions q_{BC} are evaluated prior to integrate eq.(3), and the implementation of the rest part is straightforward except the discrete heaviside function. The space of 5 bits is enough to distinguish 30 kinds of different boundary conditions for each face of a cell. So far, a 4 bytes integer array is assigned to express all cell faces of a cell. Bit-representation is employed and an ID that identifies a kind of BCs is encoded before the calculation. An integer arithmetic shift operation is possible to operate efficiently with coupling the SIMD computation unit and is expected at high performance. In fig. 1, for instance, encoding can be written by `b |= (0x3 <<(5*(j-1)))`, j=1 $\sim$ 6, where 3 is ID and j corresponds to the direction of west, east, south, north, bottom and top, respectively. Naïve implementation requires the memory space of 4 bytes integer times six, on the other hand, our implementation is one-sixth compared to the naïve version. In addition to save memory space, the proposed implementation will generate a high performance code. Williams[1] showed a Roofline model that offers insight on how to improve the performance of software and hardware. The proposed implementation is able to reduce the amount of load and store in a do-loop. Therefore the bit-representation scheme leads promising low Byte/Flop code to avoid memory-boundness.

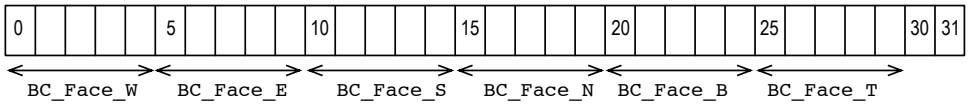

Figure 1: Compressed bit flag to represent the kind of BCs.

Preliminary test case showed that present implementation achieved more than twice performance on the K-computer and other commodity clusters. Besides, it was found that performance degradation is suppressed when arbitrary boundary conditions are employed inside a computational domain.

REFERENCES

[1] S. Williams, A. Waterman and D. Patterson. Roofline: An Insightful Visual Performance Model for Multicore Architectures. *Commun. ACM*, Vol.**52**, No.**4**, 65–76, 2009.