Implicit Unstructured Computational Aerodynamics on Many-Integrated Core Architecture

Mohammed A. Al Farhan and David E. Keyes
Extreme Computing Research Center, King Abdullah University of Science and Technology

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

Abstract

This research aims to understand the performance of PETSc-FUN3D, a fully nonlinear implicit unstructured grid incompressible or compressible Euler code with origins at NASA and the U.S. DOE, on many-integrated core architecture and how a hybrid-programming paradigm (MPI+OpenMP) can exploit Intel Xeon Phi hardware with upwards of 60 cores per node and 4 threads per core. For the current contribution, we focus on strong scaling with many-integrated core hardware. In most implicit PDE-based codes, while the linear algebraic kernel is limited by the bottleneck of memory bandwidth, the flux kernel arising in control volume discretization of the conservation law residuals and the preconditioner for the Jacobian exploits the Phi hardware well.

Motivation

Recent studies of PETSc-FUN3D [1, 2] investigated the performance of multithreading for shared memory for the modest numbers of threads per node available at the time. They show the potential for hybrid-programming paradigm compared to the pure MPI case. The execution time of the flux kernel is considerably improved by the multithreading approach. Algorithmically, this result in the performance is due to the improvement in the linear system solver since the parallel single-level additive Schwarz preconditioner is stronger with the hybrid case because of the usage of larger subdomains. The results derived from the foregoing would motivate more thorough investigation of the applicability and amenability of the flux kernel of PETSc-FUN3D on many-integrated core hardware with the potential to run hundreds of threads simultaneously within a node.

Flux Evaluation Kernel

The flux kernel reads the flow variables that are stored in the nodes (vertex-centered code) at each iteration that traverses over the node-based array to update residual values at each node of the edges, see Fig 1.

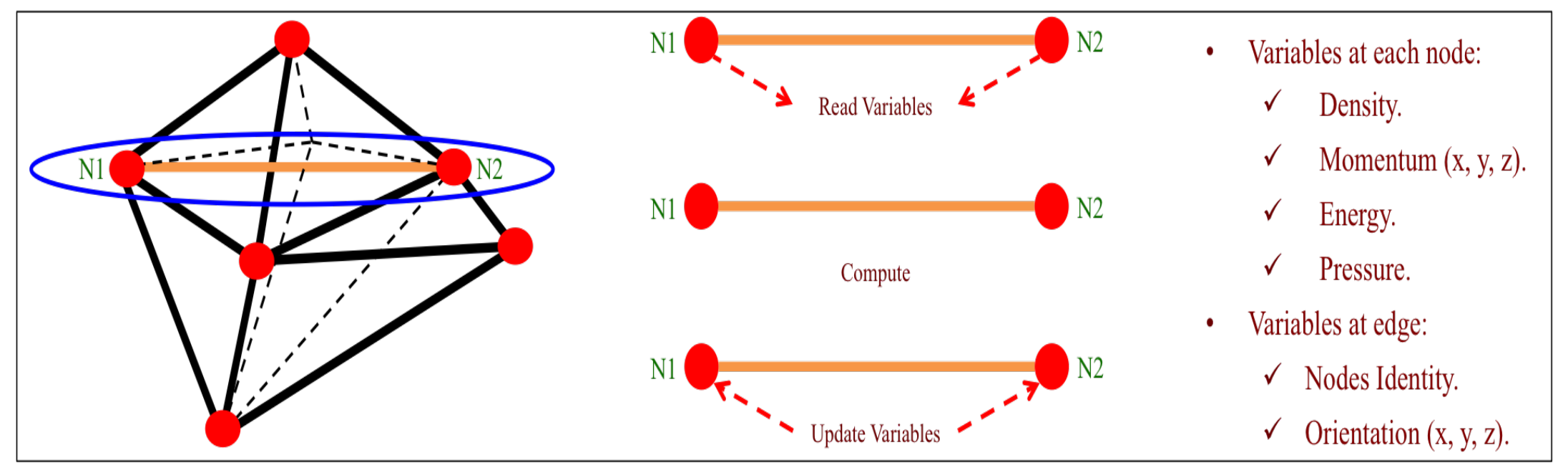

Figure 1: Flux computational kernel

Xeon Phi at a Glance

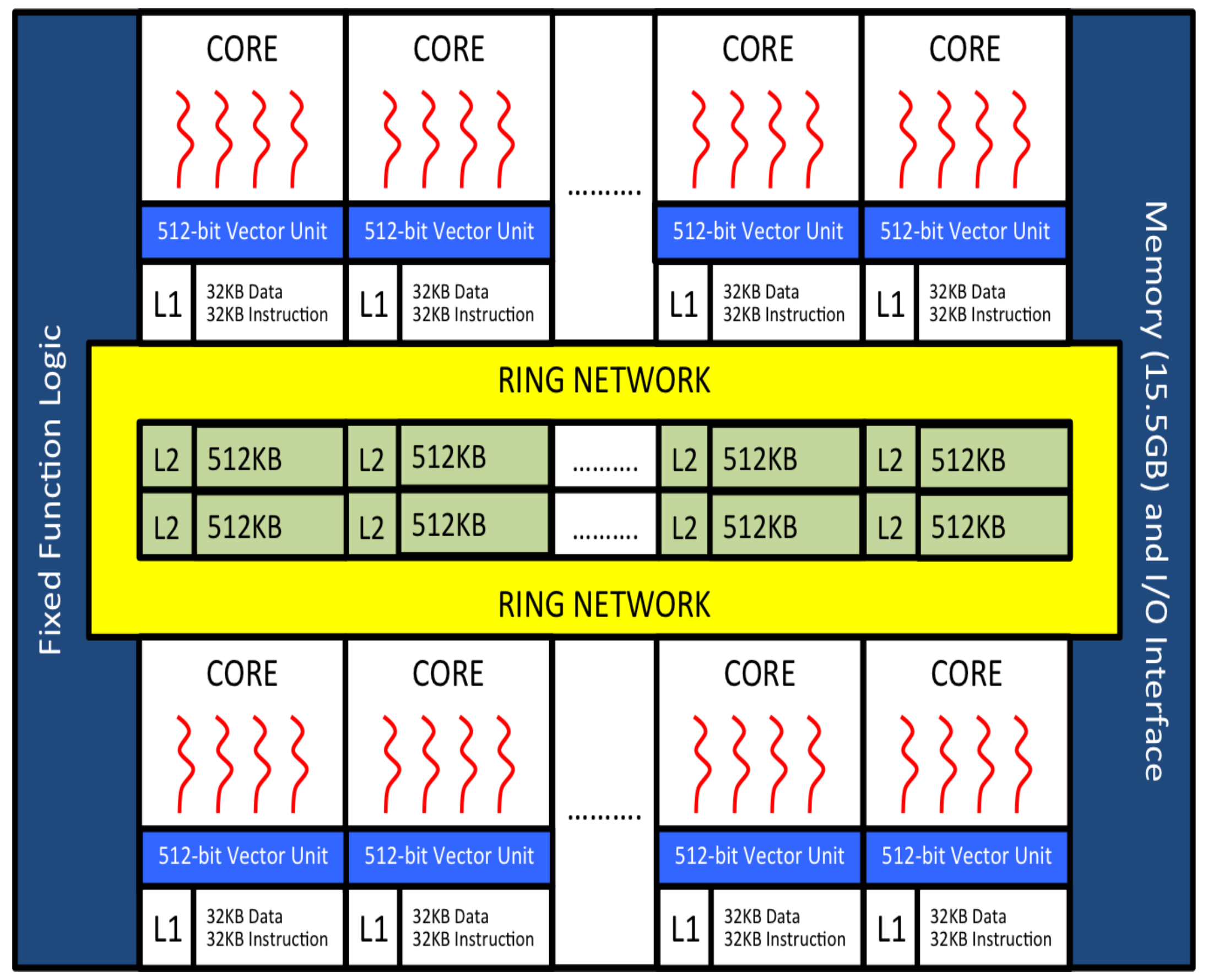

Figure 2: Intel Xeon Phi architecture

Intel Xeon Phi has 61 cores with 512-bit SIMD vectors. Each core runs 4 threads and has two 32KB L1 caches. Also, the L2 caches (512KB) of all cores are interconnected via a bidirectional ring bus, see Fig 2. Every core has a short in-order pipeline which ensures that the overhead for branch misprediction is negligible.

Results and Discussion

In our preliminary demonstration, we offload the flux kernel to Intel Xeon Phi chip as a coprocessor in two different ways, and then we examine the running time of different numbers of OpenMP threads with different affinities (thread distribution) between the cores. Each of these produces different performance results that depend upon the spatial and temporal locality of data on L1 and L2 caches. The results (shown in Fig 3) show that the flux kernel is amenable to a dynamic number of threads with careful attention to the subdivision of each subdomain.

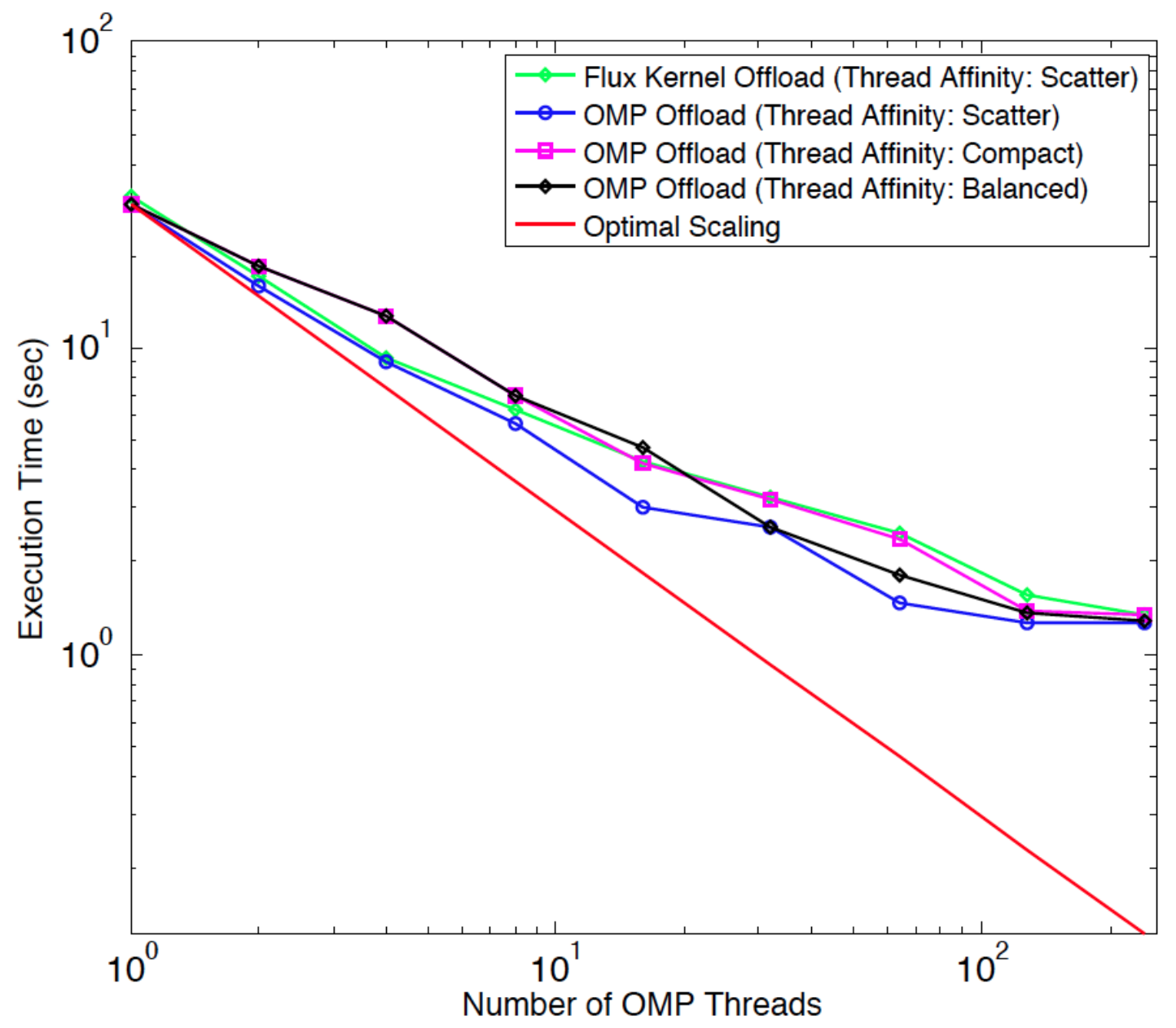

Figure 3: Strong thread scaling for offloading the flux kernel onto the Intel Xeon Phi

Also, they show that the best performance is achieved using a scatter distribution, where threads are spread between all cores in the chip (using a round-robin placements), see Fig 4, by improving the locality of data in caches, while also reducing the number of unused cores.

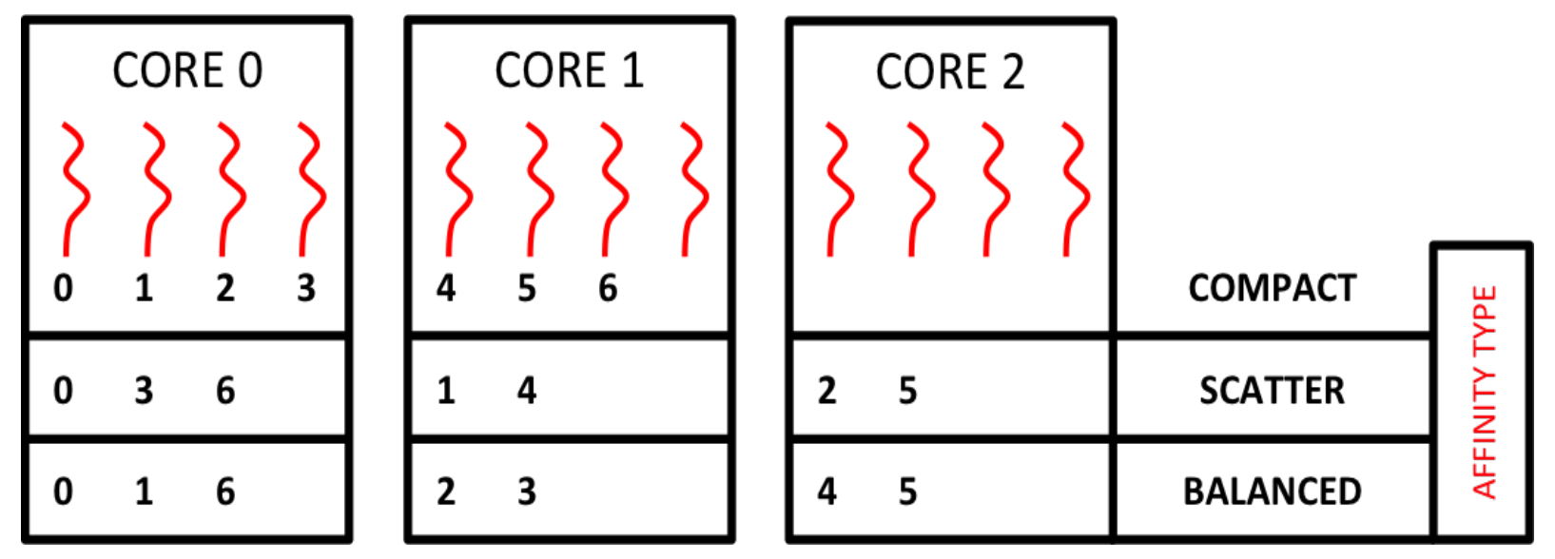

Figure 4: Thread distribution between the cores (numbers represent threads' IDs)

Conclusion

Despite the gains derived from the foregoing, the downside of using Intel Xeon Phi or any other acceleration hardware, is the data transmission back-and-forth between the host (CPU) memory and the device (coprocessor) memory. This is because of the link (PCI Express bus) bandwidth limitation, which drastically inhibits the performance of the application. The data movement techniques in such system can be the "make-or-break" feature of the implementation, therefore, further tuning of the Phi implementation of PETSc-FUN3D placement is required to achieve the best performance.

Acknowledgment

We are very appreciative of the collaborations by Dinesh Kaushik of Argonne National Laboratory, as well as Anand Deshpande and Dheevatsa Mudigere of Intel Parallel Computing Lab in Bangalore, India.

References

[1] W. D. Gropp, D. K. Kaushik, D. E. Keyes, and B. F. Smith. High performance parallel implicit CFD. *Parallel Computing*, 2001.

[2] D. K. Kaushik, D. E. Keyes, S. Balay, and B. F. Smith. Hybrid programming model for implicit pde simulations on multicore architectures. In *OpenMP in the Petascale Era: 7th International Workshop on OpenMP (IWOMP)*, volume 6665 of *Lecture Notes in Computer Science*, pages 12–21, Chicago, IL, USA, June 2011. Springer Berlin Heidelberg.

Thuwal 23955-6900, Saudi Arabia, {mohammed.farhan, david.keyes}@kaust.edu.sa