KNL Performance Comparison: *UCNS3D*

April 2017

1. Compilation, Setup and Input

Compilation

The compilation flags for both systems as well as their version of compilers is listed below:

Compiler/Flags	ARCHER XEON	ARCHER KNL
Intel Compiler	15.0.2.164	17.0.0.098
Flags	-i4 -r8 -O3 -ipo -xHost -fp model precise	-i4 -r8 -O3 -xmic-avx512 -ipo -fp-model precise -qopenmp -qopenmp-link=static

Setup

The Taylor Green vortex Implicit Large Eddy Simulation (ILES) test case was used along with a WENO 6th-order Finite-volume scheme, a 4th order Runge-Kutta SSP time stepping scheme, and HLLC Riemann solver with 21 Gaussian quadrature points per triangular face, and 35 Gaussian quadrature points per tetrahedral element for volume integrals. (REF1, REF2, REF3 for details of methods)
KNL setup: All the simulations where run on the quad100 nodes, with pure MPI, and MPI+OMP, with the OMP SCHEDULE GUIDED for all the parallel loops. The ideal configuration was found to be when using 4 MPI processes and 64 OMP threads per MPI process, with 4 hyperthreads. The single threaded version of intel MKL was used for all the DGEMM computations of the least-squares reconstructions performed.
Ivy Bridge setup: All the simulations where run using only MPI, without any OMP or hyperthreads using 24 MPI processes per core.

Simulations were run three times for 100 time steps, and the maximum computational time per MPI process was computed, and an average was made to get the simulation time per time step.

Input

Three different grid sizes where used consisting of unstructured tetrahedral elements of 0.05, 0.2 and 0.8 Million cells respectively.

UCNS3D code references: (website in development)
REF1: http://www.sciencedirect.com/science/article/pii/S0045793017300014
REF2: http://www.sciencedirect.com/science/article/pii/S0021999110006388
REF3:http://www.sciencedirect.com/science/article/pii/S0021999113006062

2. Performance Data

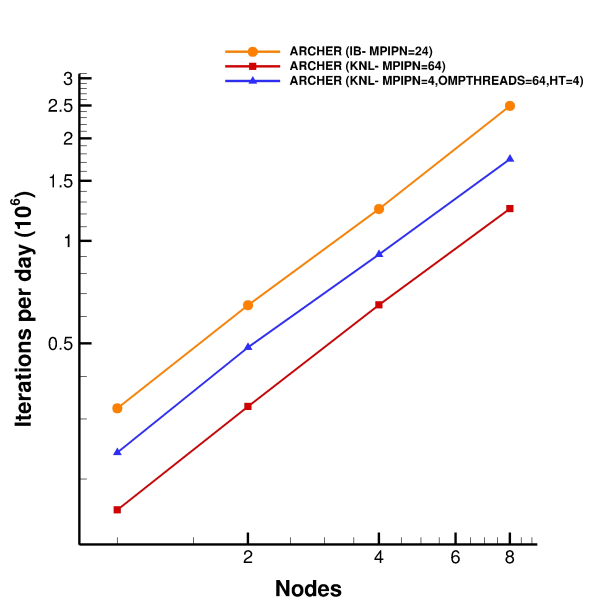

Fig1: The plot shows a performance comparison between KNL and Xeon system for the Taylor Green Vortex simulations with 0.05 Million cells grid.

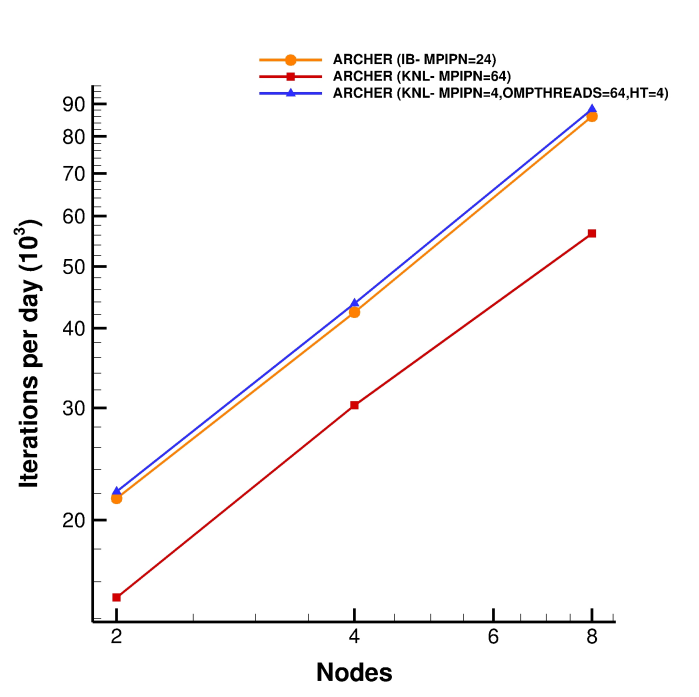

Fig2: The plot shows a performance comparison between KNL and Xeon system for the Taylor Green Vortex simulations with 0.2 Million cells grid.

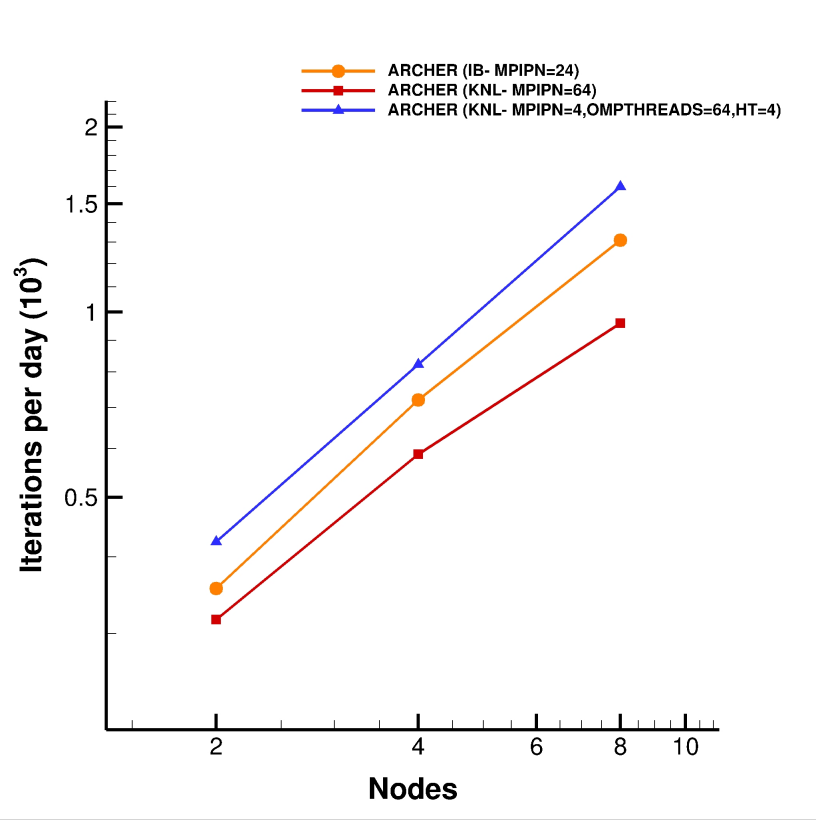

Fig3: The plot shows a performance comparison between KNL and Xeon system for the Taylor Green Vortex simulations with 0.8 Million cells grid.

Table1: 0.05 Million cells results, (Time in Millions of iterations per day)

Nodes	Archer (IB, MPIPN=24)	Archer (KNL, MPIPN=64)	Archer (KNL, MPIPN=4, OMP=64,HT=4)	Speedup (KNL MPI-OMP/IB)
1	0.3231	0.1627	0.2398	0.742
2	0.597	0.30101	0.44864	0.751
4	1.088	0.564	0.804	0.738
8	2.464	1.231	1.723	0.699

Table2: 0.2 Million cells results, (Time in Thousands of iterations per day)

Nodes	Archer (IB, MPIPN=24)	Archer (KNL, MPIPN=64)	Archer (KNL, MPIPN=4, OMP=64,HT=4)	Speedup (KNL MPI-OMP/IB)
2	21.7	15.164	22.240	1.02
4	42.66	30.44	44.08	1.03
8	85.62	56.03	87.79	1.02

Table3: 0.8 Million cells results, (Time in Thousands of iterations per day)

Nodes	Archer (IB, MPIPN=24)	Archer (KNL, MPIPN=64)	Archer (KNL, MPIPN=4, OMP=64,HT=4)	Speedup (KNL MPI-OMP/IB)
2	0.359	0.319	0.427	1.18
4	0.717	0.586	0.820	1.14
8	1.305	0.957	1.594	1.22

3. Summary and Conclusions

We were excited with the ease of portability under this architecture, and to the best of our knowledge this is the first time that a very high-order finite-volume CFD code for unstructured meshes has been ported in this architecture. The results obtained indicate that for the medium and large grid sizes the UCNS3D on the new KNL system is 2-22% faster compared to the Xeon Ivybridge processors.

Following the PRACE project "HOVE" where the present UCNS3D code has been optimized by rewriting algorithms, employing the DGEMM BLAS mkl libraries for matrix multiplications where the majority of the time is spent, we would further focus on improving the vectorization aspects of the code although the non-contiguous memory access of an unstructured CFD code is not assisting in this direction.

We would like to see a larger system using this architecture, where production runs of larger scale problems could be feasible. The main reason that we expect to see additional significant benefits are firstly the reduced memory footprint of the code when using MPI+OMP under these architectures, high-memory bandwidth beneficial for non-contiguous memory access patterns, and finally it is the ideal platform for performing high-fidelity unsteady turbulent calculations (ILES) with very high-order schemes due to the large ratio of computational work to communication work.