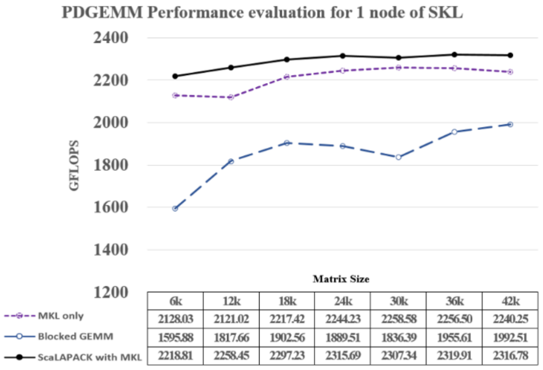
Evaluating Parallel Double-precision General Matrix Multiplication based on Blocked GEMM Algorithm on Intel Skylake clusters

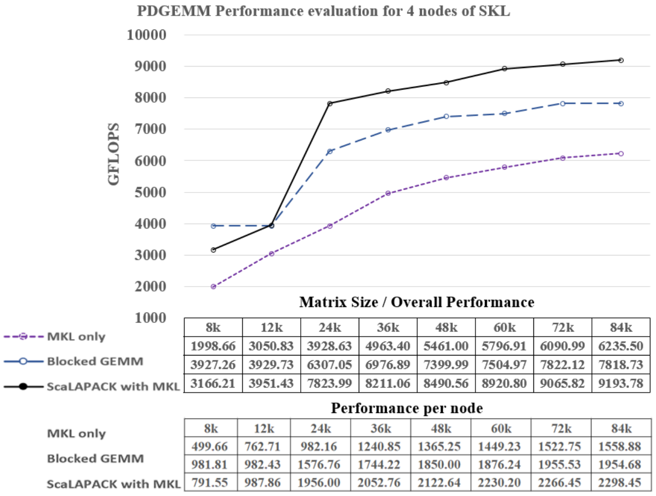
Yoosang Park, Thi My Tuyen Nguyen, and Jaeyoung Choi*
School of Computer Science & Engineering, Soongsil University
369 Sangdo-Ro, Dongjak-Gu, Seoul 156-743, Republic of Korea
Corresponding author (choi@ssu.ac.kr)

In high-performance computing, General Matrix-Matrix Multiplication routine (xGEMM) is crucial factor when systems deal with multiplications of matrices. Because modern applications use massive datasets which are equivalently represented in a matrix form. However, as increasing matrix sizes that are loaded on computing systems, the complexities of time consuming and calculation density relatively raises costs. It can be arranged by using programs in parallel computing environments, which processing units are physically and logically linked together in clusters [1]. Recent demands for domains of processing massive datasets indicate that computing workloads can be efficiently done by applying the high-performance computing techniques [2]. Thus, the core concepts mainly aim to how matrices can be handled for the specific computer architecture, and how broadcasting methods are applied during the multiplications.

As one of choices for matrix-matrix multiplication methods that we can try, the research of Blocked-GEMM algorithm for the Intel Knights Landing processor (KNL) has been previously conducted for processing double-precision GEMM (DGEMM) [3]. The research shows that the implementation uses the block matrices which fit in multi-levels of cache on the given architecture. And it helps sub-matrices to remain on the area of desired cache and register for less cache-miss occurrences on the calculations. The current research aims this algorithm to fit into Intel Skylake (SKL) clusters. It is shown that the blocked GEMM algorithm in SKL for scalable environments performs in the middle of between a naïve Intel Math Kernel Library (MKL) approach and a combination of ScaLAPACK with MKL attached. Future works yet to be done include that the adjustments should be improved for SKL clusters considering two major factors of achieving calculation performances and communication costs on each node.


PDGEMM Performance evaluation for 1 node of SKL

	6k	12k	18k	24k	30k	36k	42k
MKL only	2128.03	2121.02	2217.42	2244.23	2258.58	2256.50	2240.25
Blocked GEMM	1595.88	1817.66	1902.56	1889.51	1836.39	1955.61	1992.51
ScaLAPACK with MKL	2218.81	2258.45	2297.23	2315.69	2307.34	2319.91	2316.78


PDGEMM Performance evaluation for 4 nodes of SKL

Matrix Size / Overall Performance

	8k	12k	24k	36k	48k	60k	72k	84k
MKL only	1998.66	3050.83	3928.63	4963.40	5461.00	5796.91	6090.99	6235.50
Blocked GEMM	3927.26	3929.73	6307.05	6976.89	7399.99	7504.97	7822.12	7818.73
ScaLAPACK with MKL	3166.21	3951.43	7823.99	8211.06	8490.56	8920.80	9065.82	9193.78

Performance per node

	8k	12k	24k	36k	48k	60k	72k	84k
MKL only	499.66	762.71	982.16	1240.85	1365.25	1449.23	1522.75	1558.88
Blocked GEMM	981.81	982.43	1576.76	1744.22	1850.00	1876.24	1955.53	1954.68
ScaLAPACK with MKL	791.55	987.86	1956.00	2052.76	2122.64	2230.20	2266.45	2298.45

Figure 1. PDGEMM performance evaluations on SKL clusters: 1 node (left) and 4 nodes (right)

Acknowledgments This work was supported by the Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Korea government (MSIT) (NRF-2015M3C4A7065662).

References

[1] D. Oliveira *et al.*, "Experimental and analytical study of Xeon Phi reliability," *Proceedings of the International Conference for High-Performance Computing, Networking, Storage and Analysis (SC '17)*, No. 28, pp. 1-12. (2017).

[2] H. Anzt *et al.*, "Bringing High Performance computing to big data Algorithms," In *Handbook of Big Data Technologies, Springer,* pp. 777-806. (2017).

[3] R. Lim *et al.* "An implementation of matrix-matrix multiplication on the Intel KNL processor with AVX-512," In Cluster Computing, Vol. 21, No. 4, pp.1785-1795. (2018).