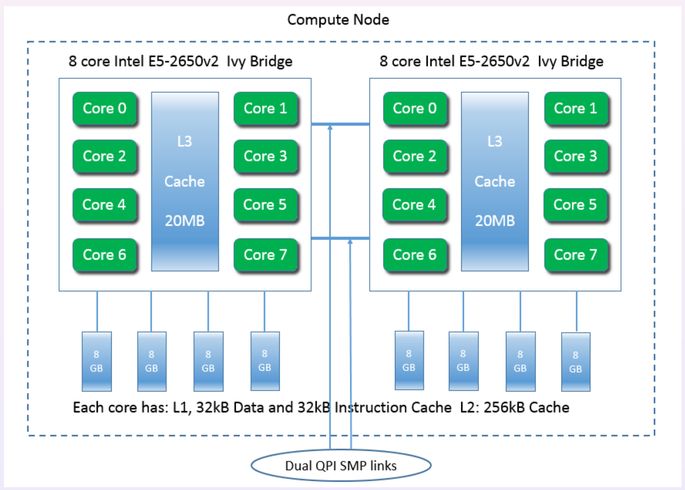
Pushing the Limits of the Maya Cluster

UMBC REU Site: Interdisciplinary Program in High Performance Computing
Adam Cunningham[1], Gerald Payton[2], Jack Slettebak[1], Jordi Wolfson-Pou[3]
Graduate assistants: Jonathan Graf[2], Xuan Huang[2] Samuel Khuvis[2]
Faculty mentor: Matthias K. Gobbert[2], Clients: Thomas Salter[4] and David J. Mountain[4]
[1]CSEE, UMBC, [2]Math and Stat, UMBC, [3]UC Santa Cruz, [4]Advanced Computing Systems Research Program

Benchmarking

Maya is the 240-node supercomputer in the UMBC High Performance Computing Facility (`www.umbc.edu/hpcf`).

- The 72 newest nodes have two eight-core Intel E5-2650v2 Ivy Bridge CPUs, making a single node capable of running 16 processes/threads simultaneously.



- The nodes are connected by a high-performance quad-data rate (QDR) InfiniBand interconnect.

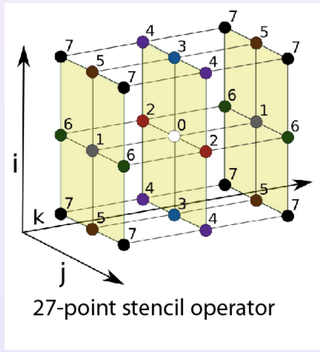
⇒ The new hardware requires testing and benchmarking to give insight into its full potential.
We report here on the High Performance Conjugate Gradient (HPCG) Benchmark developed by Sandia National Laboratories.

References

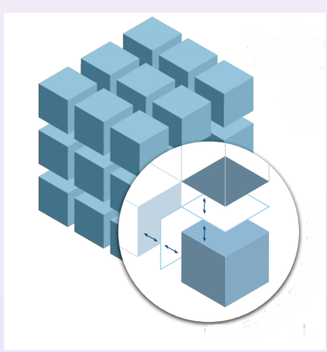
- Sandia HPCG Benchmark: `https://software.sandia.gov/hpcg/`
- For complete details and additional tests of maya, see report HPCF–2014–14 at `www.umbc.edu/hpcf` > Publications.

HPCG Benchmark

The HPCG benchmark solves the Poisson equation $-\Delta u = f$ on a three-dimensional domain $\Omega \subset \mathbb{R}^3$. A discretization on a global grid with resolution $N_x \times N_y \times N_z$ with a 27-point stencil at each grid point generates a system of linear equations with a large, sparse, highly structured system matrix. This is more relevant to many applications than the dense system matrix of the LINPACK benchmark.



27-point stencil operator



The benchmark utilizes a preconditioned conjugate gradient method to iteratively solve the system.

Experimental Design

The HPCG benchmark uses a 3-D grid of $P = p_x \times p_y \times p_z$ parallel MPI processes. We consider $P = 1, 8, 64, 512$ in our experiments.
Each process hosts a local subgrid of size $n_x \times n_y \times n_z$. Thus, $N_x = n_x\, p_x$, $N_y = n_y\, p_y$, and $N_z = n_z\, p_z$ and the total number of unknowns $N_x \times N_y \times N_z$ scales with the number of processes.
For $P = 512$ processes, the global grid ranges from millions to billions of unknowns:

$n_x \times n_y \times n_z$	$N_x \times N_y \times N_z$
$16 \times 16 \times 16$	2,097,152
$32 \times 32 \times 32$	16,777,216
$64 \times 64 \times 64$	134,217,728
$128 \times 128 \times 128$	1,073,741,824

Results

Observed GFLOP/s for HPCG Revision 2.4 with execution time 60 using the Intel C++ compiler and MVAPICH2. Local subgrid dimensions $n_x \times n_y \times n_z$ and fixed $P = 512$ parallel MPI processes (N nodes with p_N processes per node) and n_t OpenMP threads per MPI process. Possible combinations are $N = 32$ nodes with $p_N = 16$ processes per node and $n_t = 1$ thread per process or $N = 64$ with $p_N = 8$ and $n_t = 1, 2$:

$n_x = n_y = n_z = 16$		$n_t = 1$	$n_t = 2$
$N = 32$	$p_N = 16$	45.58	N/A
$N = 64$	$p_N = 8$	113.50	112.36
$n_x = n_y = n_z = 32$		$n_t = 1$	$n_t = 2$
$N = 32$	$p_N = 16$	170.03	N/A
$N = 64$	$p_N = 8$	209.92	211.84
$n_x = n_y = n_z = 64$		$n_t = 1$	$n_t = 2$
$N = 32$	$p_N = 16$	223.92	N/A
$N = 64$	$p_N = 8$	209.62	238.82
$n_x = n_y = n_z = 128$		$n_t = 1$	$n_t = 2$
$N = 32$	$p_N = 16$	233.98	N/A
$N = 64$	$p_N = 8$	210.94	230.42

- Larger problems allow for better performance, as calculation time dominates communication time.
- Increasing the number of threads per MPI process may increase computational throughput as delayes in memory access are masked by process switching.
- Optimal performance is achieved using all 16 cores per node, whether via MPI processes or via OpenMP multi-threading.

Acknowledgments

- REU Site: `www.umbc.edu/hpcreu`
- NSF, NSA, UMBC, HPCF, CIRC