The HPCchallenge Benchmark

http://icl.cs.utk.edu/hpcc/

Jack Dongarra
Piotr Luszczek

Innovative Computing Laboratory
University of Tennessee
and
Computer Science and Mathematics Division
Oak Ridge National Laboratory

High Productivity Computing Systems
Phase II Teams

Industry:

IBM — PI: Elnozahy

Sun microsystems — PI: Rulifson

CRAY — PI: Smith

Goal:

- Provide a new generation of economically viable high productivity computing systems for the national security and industrial user community (2007 – 2010)

Productivity Team (Lincoln Lead)

MIT Lincoln Laboratory — PI: Kepner

ISI

UCSD

MITRE — PI: Koester

ICL UT

Los Alamos NATIONAL LABORATORY

ARGONNE NATIONAL LABORATORY

UCSB

LCS

Ohio State

CODESOURCERY

Goal:

- Develop a procurement quality assessment methodology that will be the basis of 2010+ HPC procurements

HPCS

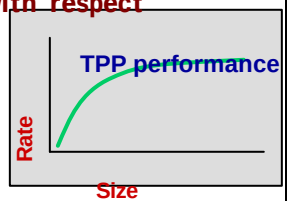
HPCchallenge Benchmark

- Consists of basically 5 benchmarks;
  - Think of it as a framework or harness for adding benchmarks of interest.
- HPL is the Linpack TPP benchmark. The test stresses the floating point performance of a system. (Gflop/s)
- STREAM is a benchmark that measures sustainable memory bandwidth (GB/s)
  - COPY, SCALE, ADD, & TRIAD
- RandomAccess measures the rate of random updates of memory. (GUp/s)
- PTRANS measures the rate of transfer for larges arrays of data from multiprocessor's memory. (GB/s)
- BW/Latency measure simple ping-pong and more complicated simulation communications. (µs, GB/s)

http://icl.cs.utk.edu/hpcc/

HPL Benchmark

- TPP Linpack Benchmark
- Used for the Top500 ratings
- Solve *Ax=b*, dense problem, matrix is random
  - Uses LU decomposition with partial pivoting
  - Based on the ScaLAPACK routines but optimized
  - The algorithm is scalable in the sense that the parallel efficiency is maintained constant with respect to the per processor memory usage
  - In double precision (64-bit) arithmetic
  - Run on all processors
  - Problem size set by user
    - These settings used for the other tests



- Requires
  - An implementation of the MPI
  - An implementation of the Basic Linear Algebra Subprograms (BLAS)
- Reports total TFlop/s achieved for set of processors

http://icl.cs.utk.edu/hpcc/

STREAM Benchmark

- The STREAM Benchmark is a standard benchmark for the measurement of computer memory bandwidth
- Measures bandwidth sustainable from standard operations -- not the theoretical "peak bandwidth" provided by most vendors
- Four operations
  - COPY, SCALE
  - ADD, TRIAD
- Measures:
  - Machine Balance - relative cost of memory accesses vs arithmetic
  - Vector lengths chosen to fill local memory
- Tested on a single processor
- Tested on all processors in the set in an "embarrassingly parallel" fashion
- Reports total GB/s achieved per processor

name	kernel	bytes/iter	FLOPS/iter
COPY:	a(i) = b(i)	16	0
SCALE:	a(i) = q*b(i)	16	1
SUM:	a(i) = b(i) + c(i)	24	1
TRIAD:	a(i) = b(i) + q*c(i)	24	2

http://icl.cs.utk.edu/hpcc/

PTRANS

- Implements parallel matrix transpose
  - $A = A + B^T$
- The matrices *A* and *B* are distributed across the processors
  - Two-dimensional block-cyclic storage
    - Same storage as for HPL
- Exercises the communications pattern where pairs of processors communicate with each other simultaneously.
  - Large (out-of-cache) data transfers across the network
- Reports total GB/s achieved for set of processors

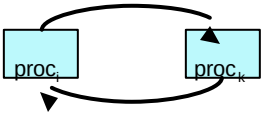
http://icl.cs.utk.edu/hpcc/

Random Access

- Integer Read-modify-write to random address
  - No spatial or temporal locality
  - Measures memory latency or the ability to hide memory latency
- Architecture stresses
  - Latency to cache and main memory
- Three forms for test
  - Tested on a single processor
  - Tested on all processors in the set in an "embarrassingly parallel" fashion
  - Tested with an MPI version across the set of processors
    - Each processor caches updates then all processors perform MPI all-to-all communication to perform updates across processors
- Reports Gup/s (Giga updates per second) per processor

http://icl.cs.utk.edu/hpcc/

Bandwidth and Latency Tests

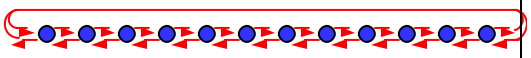


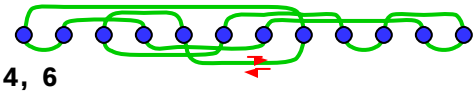
- Ping-Pong test between pairs of processors
  - Send a message from $proc_i$ to $proc_k$ then return message from $proc_k$ to $proc_i$
    - $proc_i$ MPI_Send() - $proc_k$ MPI_Recv()
    - $proc_i$ MPI_Recv() - $proc_k$ MPI_Send()
    - Other processors doing MPI_Waitall()
  - time += MPI_Wtime()
  - time /= 2
  - The test is performed between as many possible distinct pairs of processors.
    - There is an upper bound on the time for the test
  - Tries to find the weakest link amongst all pairs
    - Minimum bandwidth
    - Maximum latency
    - Not necessarily the same link will be the worst for bandwidth and latency
  - Message 8B used for latency test; take max time
  - Message 2MB used for bandwidth test; take min GB/s

http://icl.cs.utk.edu/hpcc/

Bandwidth/Latency Ring Tests (All Procs)

- **Two types of rings:**
  - **Naturally ordered**
    - **(use MPI_COMM_WORLD): 0,1,2, ... P-1.**
  - **Randomly ordered (30 rings tested)**
    - **eg.: 7, 2, 5, 0, 3, 1, 4, 6**
  - **Each node posts two sends (to its left and right neighbor) and two receives (from its left and right neighbor).**
    - **Two types of communication routines are used: combined send/receive and non-blocking send/receive.**
      - MPI_Sendrecv(TO: right_neighbor,FROM: left_neighbor)
      - MPI_Irecv(left_neighbor)MPI_Irecv(right_neighbor) and MPI_Isend(right_neighbor)MPI_Isend(left_neighbor)
    - **The smaller (better) time for each is taken (which one is smaller depends on the MPI implementation).**
  - **Message 8B used for latency test;**
  - **Message 2MB used for bandwidth test;**





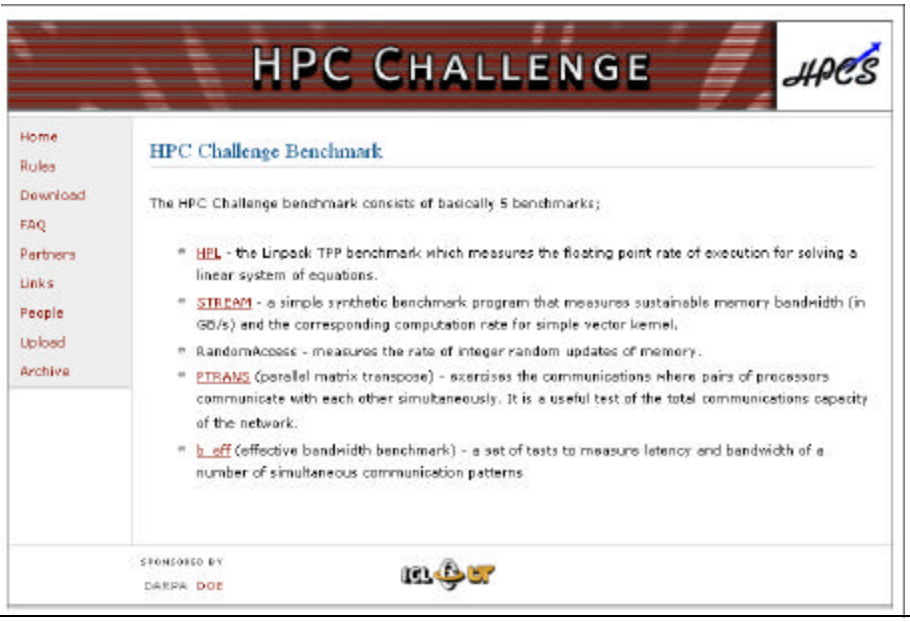
http://icl.cs.utk.edu/hpcc/

How Will This Work?

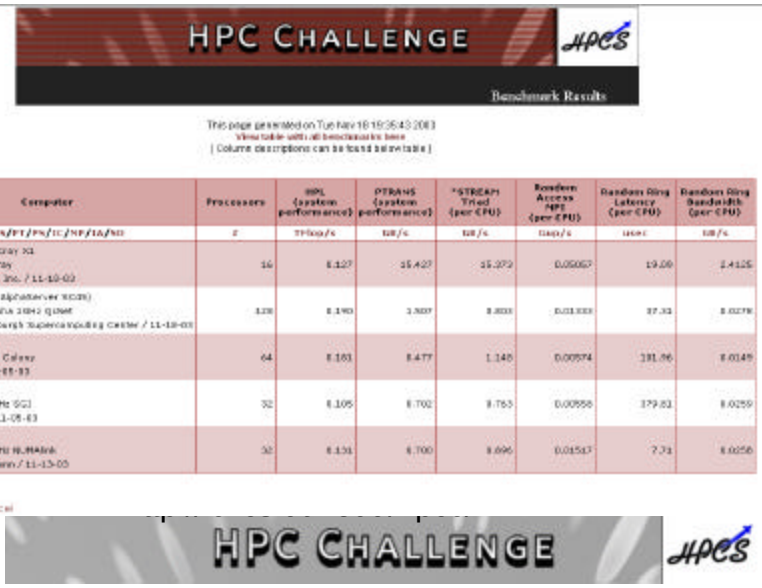
- **Single program to download and run**
  - **Simple input file similar to HPL input**
- **Base Run and Optimization Run**
  - **Base run must be made**
    - **User supplies MPI and the BLAS**
  - **Optimized run allowed to replace certain routines**
    - **User specifies what was done**
- **Results upload via website**
- **html table and Excel spreadsheet generated with performance results**
  - **No ranking**

http://icl.cs.utk.edu/hpcc/

http://icl.cs.utk.edu/hpcc/



Example of Results Page



http://icl.cs.utk.edu/hpcc/

Results Continued

http://icl.cs.utk.edu/hpcc/

Expanded Set of Benchmarks

- Developing a framework for benchmarks
- Plans are to expand the benchmark collection
- Currently working on
  - DGEMM and *DGEMM
  - Sparse matrix operations
    - In collaboration with Jim Demmel, UCB
- Thinking about an I/O benchmark

http://icl.cs.utk.edu/hpcc/

Collaborators

- David Koester, MITRE
- John McCalpin, IBM, Austin
- Rolf Rabenseifner, HLRS Stuttgart
- Jeremy Kepner, MIT Lincoln Lab
- Bob Lucas, ISI/USC

- Thanks

http://icl.cs.utk.edu/hpcc/