CRAY

POWERED BY EXPERIENCE

HPCC Results

Nathan Wichmann

Benchmark Engineer

Outline

- What is HPCC?
- Results
- Comparing current machines
- Conclusions

HPCChallenge Project Goals

- To examine the performance of HPC architectures using kernels with more challenging memory access patterns than HPL (Linpack).
- To augment the Top500 list
- To provide benchmarks that bound the performance of many real applications as a function of memory access characteristics.

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

HPC Challenge Benchmark

- **HPCC is a recently introduced (Nov'03) benchmark consisting of the following six main tests:**
  - **HPL** the Linpack TPP benchmark which measures the floating point rate of execution for solving a linear system of equations.
  - **PTRANS** (parallel matrix transpose), exercises the communications where pairs of processors communicate with each other simultaneously.
  - **STREAM**, a simple synthetic benchmark program that measures sustainable memory bandwidth (in GB/s) and the corresponding computation rate for simple vector kernel.
  - **RandomAccess**, measures the rate of random integer updates of memory.
  - **^{b}eff** (MPI bandwidth & latency test), a set of tests to measure the latency and bandwidth of a number of simultaneous communication patterns.

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

HPL - LINPACK

- **HPL**
  - Global test which utilizes the entire machine
  - Emphasizes
    - Peak Processor speed, Number of processors
- Optimized input parameters but made no code changes

HPL Results

Machine Name- # CPUS	HPL- Tflops
Cray X1- 252	2.36
Cray X1- 124	1.18
Cray T3E- 1024	0.05
HP DEC Alpha- 484	0.62
IBM Power4- 504	0.90
Linux Networx- 256	1.03

PTRANS

- PTRANS
  - Global test which utilizes the entire machine
  - Emphasizes
    - Network bandwidth and latency
- Performance is very chaotic
  - Varies dramatically depending on block size, cpu count, and problem size
  - Uses BLACs software that cannot be easily optimized
- Optimization Plans
  - Cray plans to rewrite PTRANS to directly use Co-Array Fortran or UPC
  - Expecting a significant speed increase

PTRANS Results

Machine Name- # CPUS	GB/s
Cray X1- 252	96.1
Cray X1- 124	39.4
Cray T3E- 1024	10.3
HP DEC Alpha- 484	3.74
IBM Power4- 504	5.00
Linux Networx- 256	3.11

STREAM

- Examine effects of Single CPU vs. Star CPU runs
- Cumulative STREAM TRIAD
  - Take *STREAM TRIAD number and multiply by the number of CPUs to calculate aggregate bandwidth
  - Emphasizes
    - Per CPU bandwidth under loaded conditions
    - Number of processors
- Optimizations
  - Needed to make sure arrays were aligned on cache boundaries

STREAM Results

Machine Name- # CPUS	Single CPU GB/s	Star CPU GB/s	Aggregate GB/s
Cray X1- 252	24.0	21.7	5478
Cray X1- 124	24.0	21.7	2697
Cray T3E- 1024	0.51	0.51	529
HP DEC Alpha- 484	1.66	1.38	672
IBM Power4- 504	1.99	1.71	864
Linux Networx- 256	1.64	0.77	198

Global Random Access

- Randomly generates indexes into a Global Table
  - MPI version does a local sort, an ALL to ALL, and a local gather/scatter
- Emphasizes
  - Local Gather/Scatter, Global Network bandwidth
- MPI Optimizations
  - Modified distribution of Table to eliminate if test
  - Vectorized by sorting into different bucket for each element
  - Replaced integer divide with cast and float point divide
- UPC optimization
  - Wrote version is UPC (VERY EASY)
  - Replaced integer divide with cast and float point divide

GUPS Results

Machine Name- # CPUS	GUPs	GUPs-UPC
Cray X1- 252	1.10	3.5
Cray X1- 124	1.52	
Cray T3E- 1024	0.25	
Cray X1- 60	1.31	
Cray X1- 32	1.06	
HP DEC Alpha- 484	0.45	
IBM Power4- 504	0.18	
Linux Networx- 256	0.31	

Random Ring Latency

- Your “neighbor” is a random CPU in the machine
- Take per CPU Random Ring latency number and produce a “small message per CPU bandwidth”
- Multiply that by the number of CPUs to calculate aggregate short message bandwidth
- Emphasizes
  - Scalar performance, Network Latency, # of processors
- Latency was by far the most difficult metric to interpret when comparing machines
  - Numbers vary by almost a factor if 50!!
  - How much better is 10 μsecs vs 20 μsecs vs 100 μsecs?

Ring Latency & Bandwidth Opts

- UPC Optimizations
  - Replaced MPI_Sendrecvs with equivalent UPC code
- MPI version of the ring test:

```
MPI_Sendrecv(  sndbuf_right, msglenw, MPI_LONG, right_rank,
    TO_RIGHT,  rcvbuf_left, msglenw, MPI_LONG, left_rank, TO_RIGHT,
    MPI_COMM_WORLD, &(statuses[0]) );
MPI_Sendrecv( sndbuf_left, msglenw, MPI_LONG, left_rank, TO_LEFT,
    rcvbuf_right, msglenw, MPI_LONG, right_rank, TO_LEFT,
    MPI_COMM_WORLD, &(statuses[1]) );
```

- UPC version of ring test:

```
upc_barrier;
for(i = 0; i < msglenw; i++ ){
     upc_recvbuf_left[i][right_rank] = sndbuf_right[i];
     upc_recvbuf_right[i][left_rank] = sndbuf_left[i];    }
upc_barrier;
```

Random Ring Latency Results

Machine Name- # CPUS	per CPU μsec	SM Band MB/s	UPC μsec - MB/s
Cray X1- 252	22.6	89.0	8 – 252
Cray X1- 124	20.8	47.6	8 – 124
Cray T3E- 1024	12.1	677	
HP DEC Alpha- 484	39.9	97.0	
IBM Power4- 504	367	11.0	
Linux Networx- 256	22.3	92.0	

Natural Ring Bandwidth

- Your neighbor is the next MPI process
- Take per CPU Natural Ring large message bandwidth number
- Multiply that by the number of CPUs to calculate aggregate large message bandwidth
- May not pressure the network bandwidth as much as most codes**.** Most data movement likely to be within a node and will NOT test the network.
- Emphasizes
  - Local and Network Bandwidth, Number of CPUs

Natural Ring Bandwidth Results

Machine Name- # CPUS	per CPU GB/s	LM Aggr Band GB/s	UPC GB/s - GB/s
Cray X1- 252	2.60	654.3	6 – 1512
Cray X1- 124	4.12	510.7	
Cray T3E- 1024	0.15	149.2	
HP DEC Alpha- 484	0.091	44.1	
IBM Power4- 504	0.16	79.3	
Linux Networx- 256	0.054	13.7	

Methods to Compare Machines Using HPCC

- **Normalize scores**
  - **In each category take test result and divide by the combined power of all machines**
    - **Creates a unitless number**
    - **Equal to a percentage of total power**

- **Combine all 6 unitless numbers into 1 number**
  - **Every test equal**
    - **A question of what tests are included, not how to weight each test**

HPCC: 100% HPL

Machine Name- #CPUS	Tflops
Cray X1- 252	2.35
Cray X1- 124	1.18
Linux Networx- 256	1.03
IBM Power4- 504	0.903
IBM Power4- 256	0.654
HP DEC Alpha- 484	0.618
Cray X1- 60	0.58
SGI Altix- 128	0.52

Results from ***http://icl.cs.utk.edu/hpcc/***

HPCC: Equal Weighting

Machine Name- # CPUS	HPCC Score	HPL Order
Cray X1- 252	26.5	1
Cray X1- 124	16.4	2
Cray T3E- 1024	10.2	16
Cray X1- 60	9.75	7
Cray X1- 32	6.43	10
HP DEC Alpha- 484	4.54	6
IBM Power4- 504	4.15	4
Linux Networx- 256	3.99	3

IBM Power4 256 CPU now #12; SGI Altix 128 CPU now #14

Conclusions

- The Cray X1 has superior single CPU bandwidth compared to other machines
- The Cray X1 can achieve good GUPs numbers using MPI, but it does not scale well
- The Cray X1 has very good MPI latencies when compared using a Random Ring test
  - The T3E is outstanding
  - Latency is very difficult to interpret, performance varies significantly from machine to machine
- UPC, or Co-Array Fortran, can substantially improve performance
  - Two to Three times faster
  - Much easier to code
- HPCC is a powerful new tool for examining machine performance using more challenging kernels