Experiments with Linear Algebra Operations

Jack Dongarra
INNOVATIVE COMPUTING LABORATORY

University of Tennessee
Oak Ridge National Laboratory
University of Manchester
• Autotuning
• May work well for certain computations, but
“What role will accelerators will play in the future HPC systems?”

- Conventional parallel processing vs. accelerator technologies: challenges and opportunities
- Can (and under what circumstances) systems based on other than CPU processing elements (such as FPGA, Cell, GPU) deliver performance above what is achievable on modern multiprocessors?
- The challenge of software development and programming models for effective use of accelerator technologies
- What vendors can/should do to satisfy the needs of computational scientists interested in using these architectures?
Exploiting Mixed Precision

- Current Version of the Cell has > a factor of 10 between single precision and double precision performance (204 GFlop/s to 14 GFlop/s)!
  - Next version this will narrow to a factor of 2 (as in most common processors today)
- We became interested in looking for ways to exploit the speed of SP but still retain the accuracy of DP.

SPE ~ 25 Gflop/s peak
Moving Data Around on the Cell

Worst case memory bound operations (no reuse of data)
3 data movements (2 in and 1 out) with 2 ops (SAXPY)
For the cell would be 4.6 Gflop/s (25.6 GB/s*2ops/12B) in SP.
Linear Algebra Iterative Refinement

- Exploit 32 bit floating point as much as possible.
  - Especially for the bulk of the computation
- Correct or update the solution with selective use of 64 bit floating point to provide a refined results
- Intuitively:
  - Compute a 32 bit result,
  - Calculate a correction to 32 bit result using selected higher precision and,
  - Perform the update of the 32 bit results with the correction using high precision.
IBM Cell 3.2 GHz, $Ax = b$

8 SGEMM (Embarrassingly Parallel)

Matrix Size

GFlop/s

0 50 100 150 200 250 300 350 400 450

0 50 100 150 200 250 300 350 400 450

.30 secs

3.9 secs
IBM Cell 3.2 GHz, $Ax = b$

- 8 SGEMM (Embarrassingly Parallel)

- SP Peak (204 Gflop/s)
- SP $Ax=b$ IBM
- DSGESV
- DP Peak (15 Gflop/s)
- DP $Ax=b$ IBM

- Matrix Size
- GFlop/s

- 8.3X
- 3.9 secs
- .47 secs
- .30 secs
PS3 Hardware Overview

- PowerPC
- 25.6 Gflop/s
- 25.6 Gflop/s
- 25.6 Gflop/s
- Disabled/Broken: Yield issues

- 200 GB/s injection bandwidth
- 25 GB/s between SPEs
- 3.2 GHz
- 256 MiB memory
- 256 MiB
- 32 bit peak perf 6*25.6 Gflop/s
- 153.6 Gflop/s peak
- 64 bit peak perf 6*1.8 Gflop/s
- 10.8 Gflop/s peak
- 1 Gb/s NIC
- 200 GB/s between SPEs
- GameOS
- Hypervisor

- 25.6 Gflop/s
- 25.6 Gflop/s
- 25.6 Gflop/s

- 25 GB/s

- 256 MiB

- 3.2 GHz
- 25 GB/s injection bandwidth
- 200 GB/s between SPEs
- 32 bit peak perf 6*25.6 Gflop/s
- 153.6 Gflop/s peak
- 64 bit peak perf 6*1.8 Gflop/s
- 10.8 Gflop/s peak
- 1 Gb/s NIC
- 256 MiB memory
Matrix Multiple on a 4 Node PlayStation3 Cluster

What's good

- Very cheap: ~4$ per Gflop/s (with 32 bit fl pt theoretical peak)
- Fast local computations between SPEs
- Perfect overlap between communications and computations is possible (Open-MPI running):
  - PPE does communication via MPI
  - SPEs do computation via SGEMMs

What's bad

- Gigabit network card. 1 Gb/s is too little for such computational power (150 Gflop/s per node)
- Linux can only run on top of GameOS (hypervisor)
  - Extremely high network access latencies (120 usec)
  - Low bandwidth (600 Mb/s)
- Only 256 MB local memory
- Only 6 SPEs

33 Time
Gold: Computation: 8 ms
Blue: Communication: 20 ms
SUMMA -- Model vs Measured 6 SPEs

- Model 6 SPEs
- Measures 6 SPEs

problem size vs Gflop/s graph showing the comparison between the model and measured results for 6 Single Program Elements (SPEs) on a 2x2 PlayStation3 cluster.
SUMMA on a 2x2 PlayStation3 cluster

SUMMA -- Model vs Measured 6 SPEs

memory limit

Gflop/s

problem size

Model 6 SPEs
Measures 6 SPEs
GPU Experiments

- LAPACK is running on the CPU, making calls to CUDA BLAS which are running on the GPU.
  - AMD Opteron 1.8 GHz
  - NVIDIA Quadro FX 5600
    - processors: 128 (total)
    - max performance: 346 GFlop/s SP
Performance of LAPACK LU, QR, and Cholesky with CUBLAS

AMD Opteron 1.8 GHz & NVIDIA Quadro FX 5600

Graph showing the performance of LU, QR, and Cholesky operations with matrix sizes and corresponding Gflop/s.
Collaborators / Support

Alfredo Buttari, UTK
Julien Langou, UC colorado
Julie Langou, UTK
Piotr Luszczek, MathWorks
Jakub Kurzak, UTK
Stan Tomov, UTK