High Performance Computing - Benchmarks

Dr M. Probert

http://www-users.york.ac.uk/~mijp1

Overview

- Why Benchmark?
- LINPACK
- HPC Challenge
- STREAMS
- SPEC
- Custom Benchmarks

Why Benchmark?

- How do you know which computer to buy?
  - Might be based on a thorough knowledge of the hardware specification and what all the bits mean and how well they perform
  - But what if it is new hardware?
  - And what does “how well they perform” mean?
- How do you compare two very different computers?
  - E.g. vector vs. MPP?
  - E.g. AMD vs. Intel vs. Alpha vs. IBM vs. SPARC?

Pentium 3 vs. 4

- Which was faster, a 1.2 GHz Pentium 3 or a 3 GHz Pentium 4?
- The P4 had a 31 stage instruction pipeline ('prescott' core) vs. 10 in the P3.
  - Latency of the P4 pipeline was actually higher!
  - If a section of code continuously stalled the pipeline, it would run at ~ 0.12 GFLOPS on the P3 and ~ 0.10 GFLOPS on the P4!
- Old example but principle always true – best choice of chip depends on the code!
- Benchmarks aim to give a systematic way of making comparisons based on "real world" codes.

Ranking Computers

- Top500 – a very popular list of the most powerful computers in the world
  - How are they ranked?
  - Already seen it in earlier hardware lectures
  - Based on the LINPACK benchmark
  - But what does that actually tell you?
- Need to understand what a particular benchmark *actually* measures and under what conditions
  - Then can determine whether or not this benchmark has any relevance to you and the way you intend to use that computer!

Useless Benchmarks

- Clock Speed
  - Might give some indication of relative performance within a given processor family
  - Useless between different processor families
    - My old 666 MHz Alpha EV 6/7 completed a CASTEP calculation in about the same time as a 1.7 GHz P4 Xeon!
    - Different architectures do different amount of work per clock cycle, RISC vs. CISC, etc.
  - Even between different processor generations from a given manufacturer there can be surprises
    - e.g. early Pentiums with higher clock speeds (up to 150 MHz) were slower in many "real-world" tests compared to the 486s (at 66 MHz) they were intended to replace – cache changes?
    - Ditto 1.2 GHz Pentium 3 vs 3 GHz Pentium 4

MIPS – Another Useless Benchmark

- Millions of Instructions per Second
  - (or Meaningless Indicator of Processor Speed)
  - One of the earliest indicators of speed
  - Closely related to clock speed
  - Which instruction?
    - On some architectures, a given instruction might take 20 clock cycles whereas equivalent instruction may take only 1 or 2 on a different architecture
    - What if there is no hardware support for a given instruction? CISC vs. RISC?
  - Only meaningful within a processor family, e.g. Intel used to promote the iCOMP benchmark but has now retired it in favour of industry standard benchmarks.

MFLOPS – Another Useless Benchmark

- Millions of FLoating-point Operations Per Second
  - Definition includes FP-adds and multiplies
  - What about square roots and divides? Some do it in hardware, others in microcode.
  - What about fused multiply-adds as in some CPUs? Can get multiple FLOPS per function unit per clock cycle!
  - Peak MFLOPS is pretty meaningless – very few codes will achieve anything like this due to memory performance.

- So what we need is a benchmark based upon some real-life code. Something that will combine raw CPU speed with memory performance. Something like …

LINPACK

- Actually a LINear algebra PACKage – a library not a benchmark.
  - But the developers used some of the routines, to solve a system of linear equations by Gaussian elimination, as a performance indicator
    - as the number of FLOPs required was known, the result could be expressed as average MFLOPS rate
    - LINPACK tested both floating-point performance and memory, and due to the nature of the algorithm, was seen as a “hard problem” which could not be further speeded-up – hence seen as a useful guide to real scientific code performance – hence benchmark

More LINPACK

- LINPACK test comes in various forms:
  1. 100x100 matrix, double precision, with strict use of base code, can optimise compiler flags
  2. 1000x1000 matrix, any algorithm, as long as no change in precision of answers
- But whilst in 1989 the 100x100 test was useful, the data structures were ~ 320 kB, so once cache sizes exceeded this, it became useless!

- Library lives on as LAPACK – see later lectures

LINPACK lives!

- LINPACK "Highly Parallel Computing" benchmark used as basis for Top500 ranks
  - Vendor is allowed to pick matrix size (N)
  - Information collected includes:
    - R_{peak} – system peak GFLOPS
    - N_{max} – matrix size (N) that gives highest GFLOPS for a given number of CPUs.
    - R_{max} – the GFLOPS achieved for the N_{max} size matrix
    - $N_{½}$ - matrix size that gives $R_{max}/2$ GFLOPS
  - Interest in all values – for instance, N_{max} reflects memory limitations on scaling of problem size, so high values of N_{max} and $N_{½}$ indicate system best suited to very scalable problems
  - Big computers like big problems!

Problems with LINPACK

- Very little detailed information about the networking subsystem
  - A key factor in modern cluster computers
- Hence new benchmark recently announced: the HPC Challenge benchmark
- The HPC Challenge benchmark consists of basically 7 benchmarks: a combination of LINPACK/FP tests, STREAM, parallel matrix transpose, random memory access, complex DFT, communication bandwidth and latency.

STREAM

- STREAM is memory speed benchmark (OMP)
  - Instead of trying to aggregate overall system performance into a single number, focuses exclusively on memory bandwidth
  - Measures user-sustainable memory bandwidth (not theoretical peak!), memory access costs and FP performance.
  - A *balanced system* will have comparable memory bandwidth (as measured by STREAM) to peak MFLOPS (as measured by LINPACK 1000x1000)
  - Machine balance is peak FLOPS/memory bandwidth
    - Values ~ 1 indicate a well-balanced machine – no need for cache
    - Values » 1 needs very high cache hit rate to achieve useful performance
    - Useful for all systems – not just HPC – hence popularity

- Selection from STREAM Top20 (August 2014)

Machine	Ncpu	MFLOPS	MW/s	Balance
Cray_T932 (Vector ‘96)	32	57600	44909	1.3
NEC SX-7 (Vector ‘03)	32	282419	109032	2.6
SGI Altix 4700 (ccNUMA ‘06)	1024	6553600	543771	12.1
SGI Altix UV 1000 (ccNUMA ‘10)	2048	19660800	732421	26.8
Fujitsu SPARC M10-45 (SMP ‘13)	1024	24576000	500338	49.1
Intel Xeon Phi SE10P (ACC ‘13)	61	1073600	21833	49.2

- Selection from STREAM PC-compatible (Aug 2014)

Machine	Ncpu	MFLOPS	MW/s	Balance
486-DX50 (‘95)	1	10	2.9	3.4
AMD Opteron 248 (‘03)	1/2	10666	393 / 750	11.2 / 11.7
Intel Core 2 Quad 6600 (‘07)	2/4	19200 / 38400	714 / 664	26.9 / 57.8
Intel Core2DuoE8200 DDR2/3 (‘08)	1	10666	699 / 983	15.3 / 10.9
Apple Mac-Pro (‘09)	1	10666	1119	9.5
Intel Core i7-2600 (’11)	2/4	27200 / 54400	1770 / 1722	15.4 / 31.6
Intel Core i7-4930K (‘13)	1/2/ 12	13610 * Ncpu	1912 / 2500 / … 3797	7.1 / 10.9 … 43.0

SPEC Benchmarks

- The Systems Performance Evaluation Cooperative (SPEC) is a not-for-profit industry body
  - SPEC89, SPEC92, 95 and 2000 have come and gone
  - SPEC2006 is (still) current – new v1.2 released Sept 2011
  - SPEC attempts to keep benchmarks relevant
  - Each benchmark is a mixture of C and FORTRAN codes, covering a wide spread of application areas
  - SPECmark was originally the geometric mean of the 10 codes in SPEC89 – limited scope.
  - Later versions had more codes, with some codes focusing on integer and others on FP performance, hence now get separate SPECfp2006 and SPECint2006.
  - Also a “base” version of benchmark without vendor tweaks and aggressive optimisations to stop cheating.
  - Also a “rate” version for measuring parallel throughput.
  - Additional benchmarks for graphics, MPI, Java, etc …

Sample SPEC2006 Results

Name	SPECint2006	SPECfp2006	SPECint_rate2006	SPECfp_rate2006
AMD Opteron 2356 Barcelona 2.3 GHz	13.2 base 8 cores, 2 chips	16.2 base 8 cores, 2 chips	45.6 base 4 cores, 1 chip	41.3 base 4 cores, 1 chip
Intel Core 2 Quad Q6800	20.2 base 4 cores, 1 chip	18.3 base 4 cores, 1 chip	56.2 base 4 cores, 1 chip	39.2 base 4 cores, 1 chip
Intel Core i7-975	31.6 base 4 cores, 1 chip	32.9 base 4 cores, 1 chip	121 base 4 cores, 1 chip	85.2 base 4 cores, 1 chip
IBM Power780 Power7 CPUs	29.3 base 8 cores, 1 chip	44.5 base 16 cores, 1 chip	1300 base 32 cores, 4 chips	531 base 16 cores, 2 chips

SYSmark 2014

- Another commercial benchmark, widely used in the mainstream PC industry, produced by BAPCo
  - Updated every 2 years or so until Windows Vista caused major problems – stuck at 2007 until 2011
  - Based upon typical “office” productivity and internet content creation applications
  - Useful for many PC buyers and hence manufacturers, but not for HPC

Choosing a Benchmark

- Have discussed only a small selection of the available benchmarks – see http://www.netlib.org/benchmark for more!
- Why so many?
  - No single test will tell you everything you need to know – but can get some idea by combining data from different tests as done in HPC Challenge
  - Tests become obsolete over time due to hardware developments – c.f. the LINPACK 100x100
  - And also due to software developments – particularly compilers. Once a particular benchmark becomes popular, vendors target compiler development to improve their performance on this test – hence need to regularly update & review the benchmark contents
  - Hence some industrial benchmarks keep code secret

Creating Your Own Benchmark

- Why?
  - Because the best test that is relevant to you as a HPC user, is how well your HPC codes run!
  - If you are responsible for spending a large sum (£10k - £10m) then you want to get it right!
  - Maybe your codes need special library support? Maybe your codes will test the compiler/ hardware in a non-standard way? Lots of I/O or graphics?
  - Maybe your tests will expose a bug in the compiler?

- NB Unlikely to be able to do this when buying a "standard PC"!

Making A Benchmark

- Need it to be a representative test for your needs
  - But not require too long to run (1 hour max)
  - Might require extracting a *kernel* from your code – the key computational features – and writing a simple driver.
  - Test of CPU and memory or other things
    - I/O? Graphics? Throughput? Interactive use?
- Need it to be repeatable and portable
  - Will need to average times on a given machine
  - Repeat with different compiler flags
  - Repeat on different machines
  - Automate the building/running/analysis of results?

Beware The Compiler!

- By extracting a computational kernel and stripping the code down, you may create problems
  - A clever compiler might be able to over-optimise your kernel code in a way that is not representative of the main code
  - Need to put extra obstacles in the way to confuse the compiler – not something you normally want to do!
    - E.g. executing a given loop multiple times to get a reasonably large enough time to measure may be self-defeating if the compiler can spot this and just execute the loop once!
  - Also beware the effects of cache
    - If you repeat something multiple times, the first time will incur the cache-miss cost, whilst the other iterations might all be from cache and hence run disproportionably faster! Need to flush cache between loops somehow.

Bad Benchmarks

CPU benchmark

```
# total number of flops
required is 10000
do  I = 1,10000
      x = y*z
end do
```

Any good compiler will recognise that a single trip through the loop with give the same result, and hence remove the loop entirely.

Memory benchmark

```
// repeat the benchmark
// 100 times for good
// stats
for (i=0;i<100;i++){
   t = 0.0;
   for (j=0;j<50000){
      t += A[j]*B[j];
   }
}
```

A and B may easily fit in cache. After first loop code measures cache not memory performance.

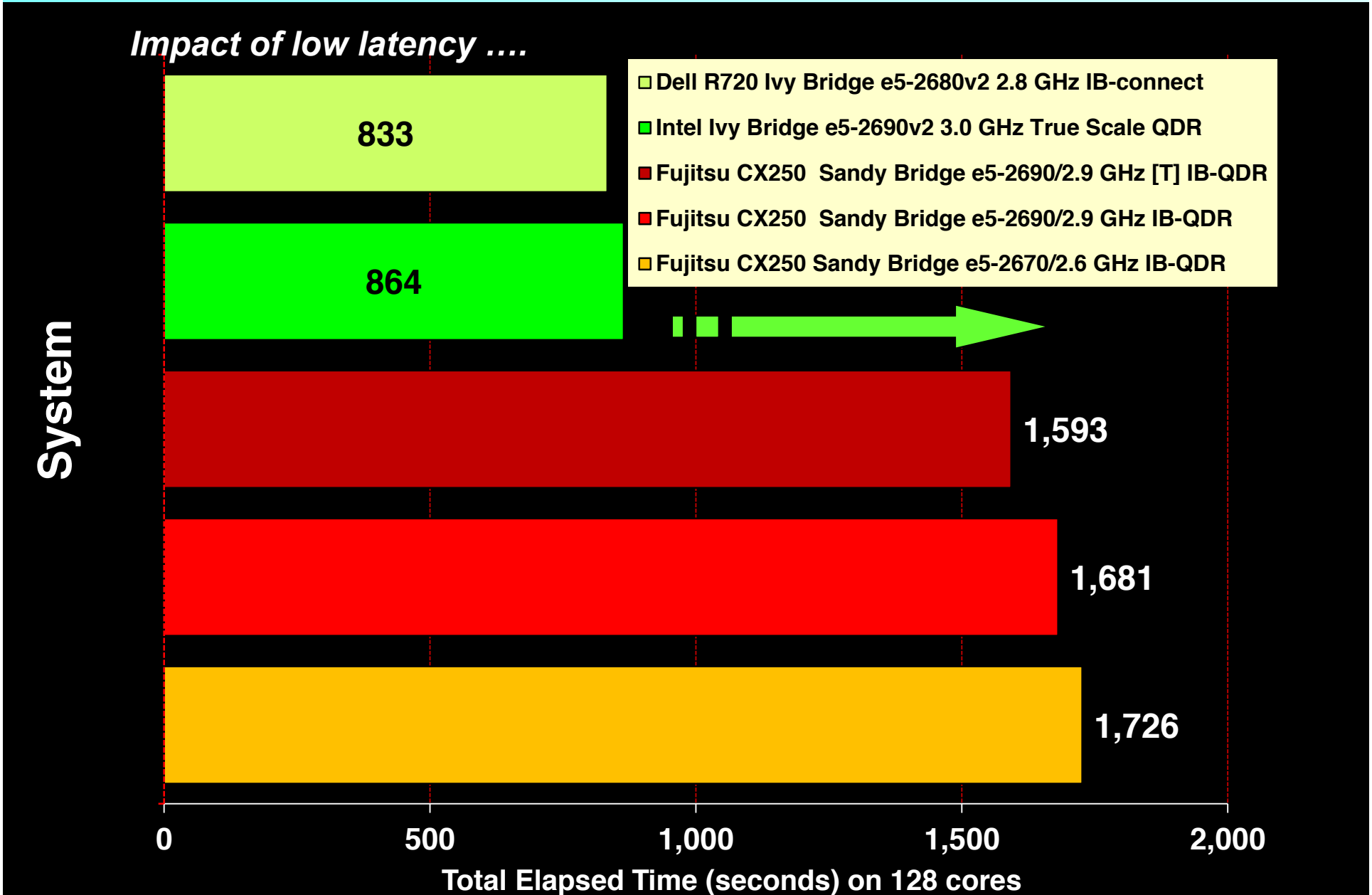
Benchmarks and Vendors

- These days, profit margins on most HPC machines are very small
  - Hence vendors are generally reluctant to give you much support in porting & running your benchmarks – would rather you stick to published data
    - i.e. the benchmarks that *they* like!
  - Not true for big computer tendering activities such as HPCx or HECToR (the sucessor to HPCx)
    - Suddenly a lot of vendor interest in porting and optimising academic codes! CASTEP was one of the benchmarks for HECToR.
  - Hence interest in events such as the Daresbury Machine Evaluation Workshop, where many HPC vendors can display their latest wares, and where users can run their own small (~15 min) benchmarks on many different systems. See: **https://eventbooking.stfc.ac.uk/news-events/mew24?agenda=1**

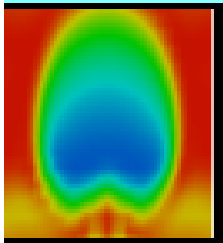
MEW24 Presentation by M. Guest

- An example of "user benchmarking" using computational chemistry kernels:
  - Uses both "synthetic" and "end-user" benchmarks
  - E.g. stripped down "kernel" or a mix of standard applications
  - Focus on multi-core systems – both CPU and whole system to look at effect of networking etc.
  - This time, focused on comparison of job with same number of cores, and also same number of nodes
  - Intel Sandy Bridge vs Ivy Bridge + different interconnects
  - LOT of details – **https://eventbooking.stfc.ac.uk/uploads/mew24/arccamartynguest.pdf**

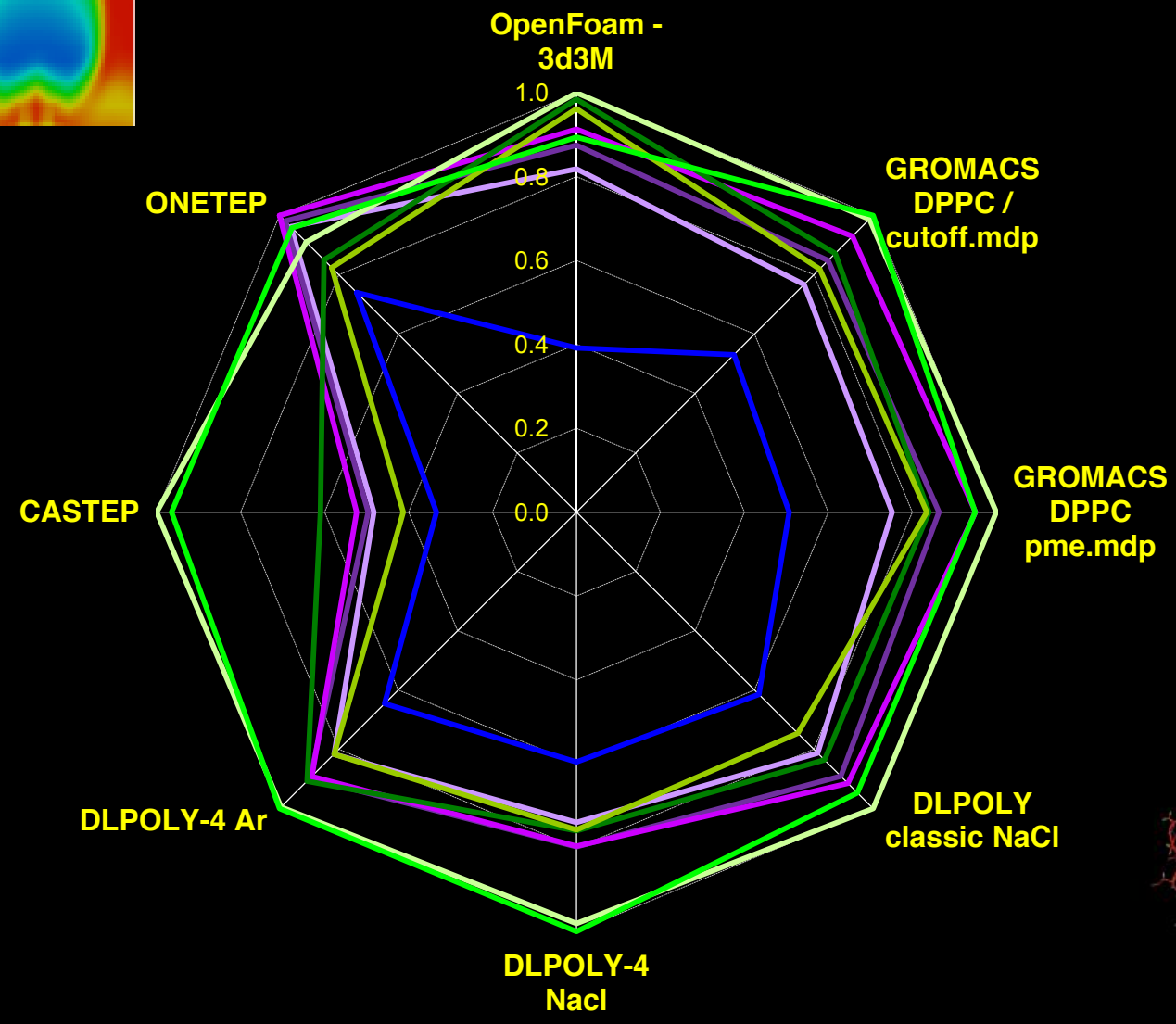
CASTEP – The MnO2 benchmark –



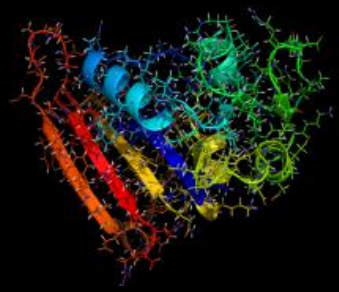
Target Codes and Data Sets – 128 PEs



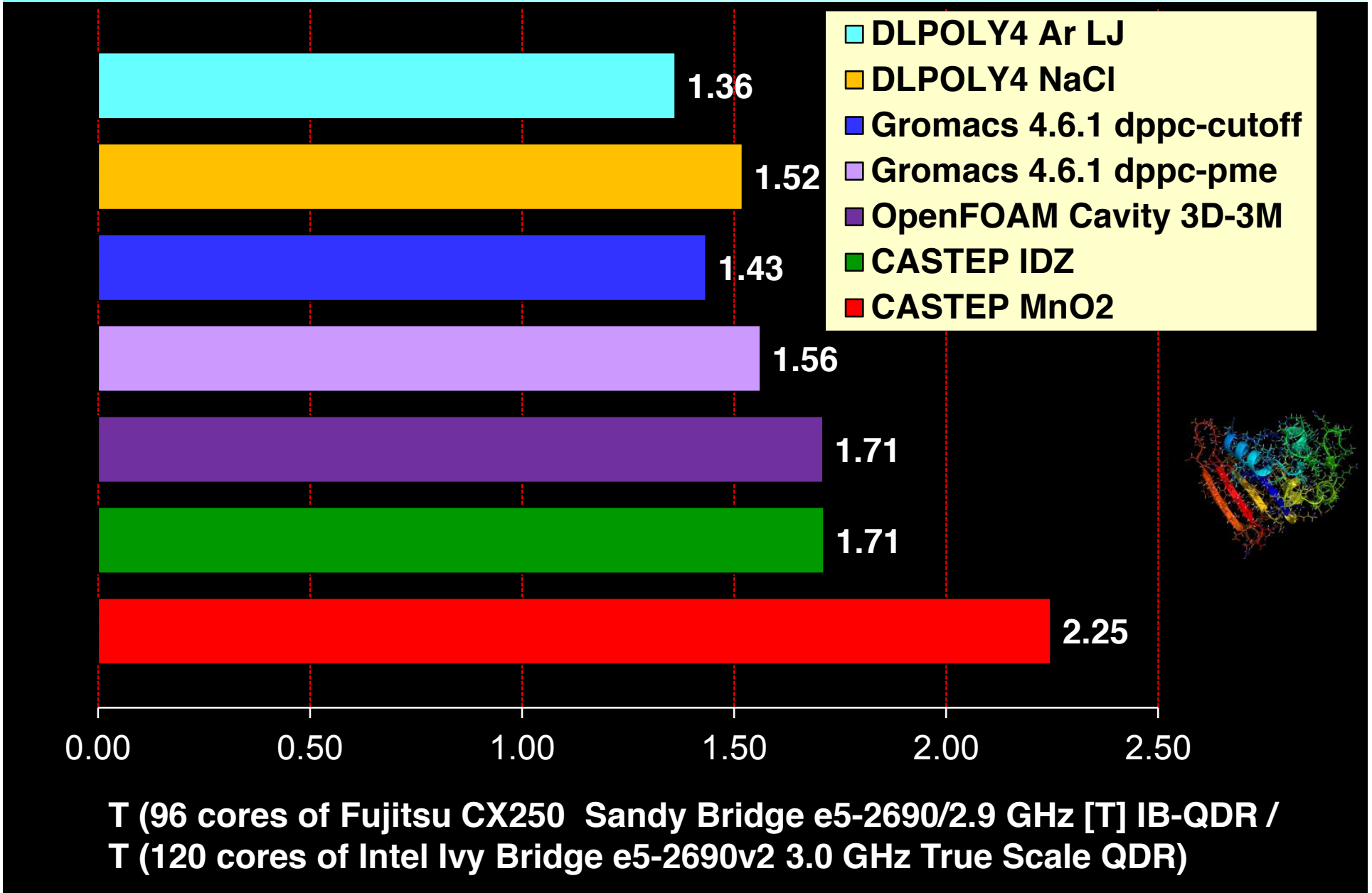
128 PE Performance [Applications]



- Fujitsu BX922 Westmere X5650 2.67GHz + IB-QDR (HTC)
- Fujitsu CX250 Sandy Bridge e5-2670/2.6 GHz IB-QDR
- Fujitsu CX250 Sandy Bridge e5-2690/2.9 GHz IB-QDR
- Fujitsu CX250 Sandy Bridge e5-2690/2.9 GHz [T] IB-QDR
- Intel Ivy Bridge e5-2697v2 2.7 GHz True Scale PSM
- Bull B710 Ivy Bridge e5-2697v2 2.7 GHz Mellanox FDR
- Dell R720 Ivy Bridge e5-2680v2 2.8 GHz IB-connect
- Intel Ivy Bridge e5-2690v2 3.0 GHz True Scale QDR



Node to Node Comparison – Six node Performance



T (96 cores of Fujitsu CX250 Sandy Bridge e5-2690/2.9 GHz [T] IB-QDR /
T (120 cores of Intel Ivy Bridge e5-2690v2 3.0 GHz True Scale QDR)

Summary

- Background to Intel's Ivy Bridge and Systems under evaluation
  - Intel's Ivy Bridge (E5-2697v2 and E5-2680v2) and Sandy Bridge-based (E5-2690 and E5-2670) Clusters
    - Bull SID cluster - E5-2697v2 with IB/FDR (2048 cores)
    - Intel True Scale clusters - Ivy Bridge dual processor nodes – both 12-way (192 cores) and 10-way clusters (640 cores)
    - 32 node Dell 720 cluster "Jupiter" at the HPC Advisory Council – 10-way Ivy Bridge processors with Mellanox Connect-IB interconnect
- Variety of Parallel Benchmarks
  - Synthetic (IMB and HPCC) and application codes
    - Six applications - DLPOLY 4 and DLPOLY Classic, Gromacs, GAMESS-UK, CASTEP, OpenFOAM
- Results from Intel Westmere, Sandy Bridge and Ivy Bridge systems
- Performance Metrics across a variety of data sets suggest that both the "Jupiter" and "*Diamond*" Ivy Bridge clusters outperform the Sandy Bridge systems

Acceptance Tests

- Having benchmarked the various machines available, it is a good idea to write an "acceptance test" into the contract for the final machine
  - When we bought Erik we specified that the machine was to be able to run CASTEP and AMBER non-stop for 8 days on 16 nodes each, without fault.
  - This was primarily to test the compilers and the thermal stability of the machine
    - Had heard "horror stories" of other clusters that would over-heat and fail/shutdown after less than 1 day of running
    - Problem eventually traced to faulty RDRAM chips
  - Same on Edred – took over 4 months to pass
    - Problems due to hardware / supplier chain / software stack

Further Reading

- Chapter 15 & 16 of “High Performance Computing (2nd edition)”, Kevin Dowd and Charles Severance, O'Reilly (1999).
- Top500 at http://www.top500.org
- LINPACK at http://www.top500.org/project/linpack
- HPC Challenge at http://icl.cs.utk.edu/hpcc
- STREAM at http://www.cs.virginia.edu/stream/
- SPEC at http://www.spec.org
- SYSmark2014 at http://www.bapco.com/products/sysmark2014/index.php
- Latest MEW at **https://eventbooking.stfc.ac.uk/news-events/mew25**