

THE UNIVERSITY *of York*

High Performance Computing - Benchmarks

Prof Matt 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. IBM vs. ...?

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 Measure

- 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 Measure

- 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 need 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
 - $N_{1/2}$ - matrix size that gives $R_{\text{max}}/2$ GFLOPS
 - Interest in all values – for instance, N_{max} reflects memory limitations on scaling of problem size, so high values of N_{max} and $N_{1/2}$ 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 the HPC Challenge benchmark:
 - a combination of LINPACK/FP tests with
 - STREAM,
 - parallel matrix transpose,
 - random memory access,
 - complex DFT,
 - communication bandwidth and latency.

STREAM

- STREAM is memory speed benchmark (OMP)
 - Instead of an aggregate overall system performance in a single number, focuses exclusively on memory bandwidth
 - Measures user-sustainable memory bandwidth (not theoretical peak!), memory access costs and FP speed.
 - *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 $\gg 1$ needs very high cache hit rate to achieve useful performance
 - Useful for all systems – not just HPC – hence popularity
 - See <http://sites.utexas.edu/jdm4372/> (the creator) for latest ...

- Selection from STREAM Top20 (last updated 2017)

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
Fujitsu SPARC M10-45 (SMP '13)	1024	24576000	500338	49.1
Intel Xeon Phi SE10P (ACC '13)	61	1073600	21833	49.2
SGI UV 3000 (ccNUMA '15)	3072	103219200	1728273	59.7

- Selection STREAM PC-compatible (2017)

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 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
Raspberry Pi _2B ('15)	1	594	70.5	8.4

SPEC Benchmarks

- The Systems Performance Evaluation Cooperative (SPEC) is a not-for-profit industry body
 - SPEC89, SPEC92, 95, 2000 and 2006 have come and gone
 - SPEC2017 still current (problems with Win Vista ...)
 - 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, Cloud, etc ...

SYSmark 2018

- 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 2014. Now cross-platform ...
 - Based upon typical “office” productivity and internet content creation applications
 - Useful for many PC buyers and hence manufacturers, but not for HPC
 - New variant for tablets in 2017, mobiles in 2018

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 software developments – particularly compilers. Vendors target compiler development to improve their performance on a 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-optimize 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 disproportionately 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 will 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 ARCHER-2
 - Suddenly a lot of vendor interest in porting and optimising academic codes! CASTEP has been one of the benchmarks since HPCx ...
 - Hence interest in events such as Computing Insights UK (formerly 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

CIUK 2021 Presentations

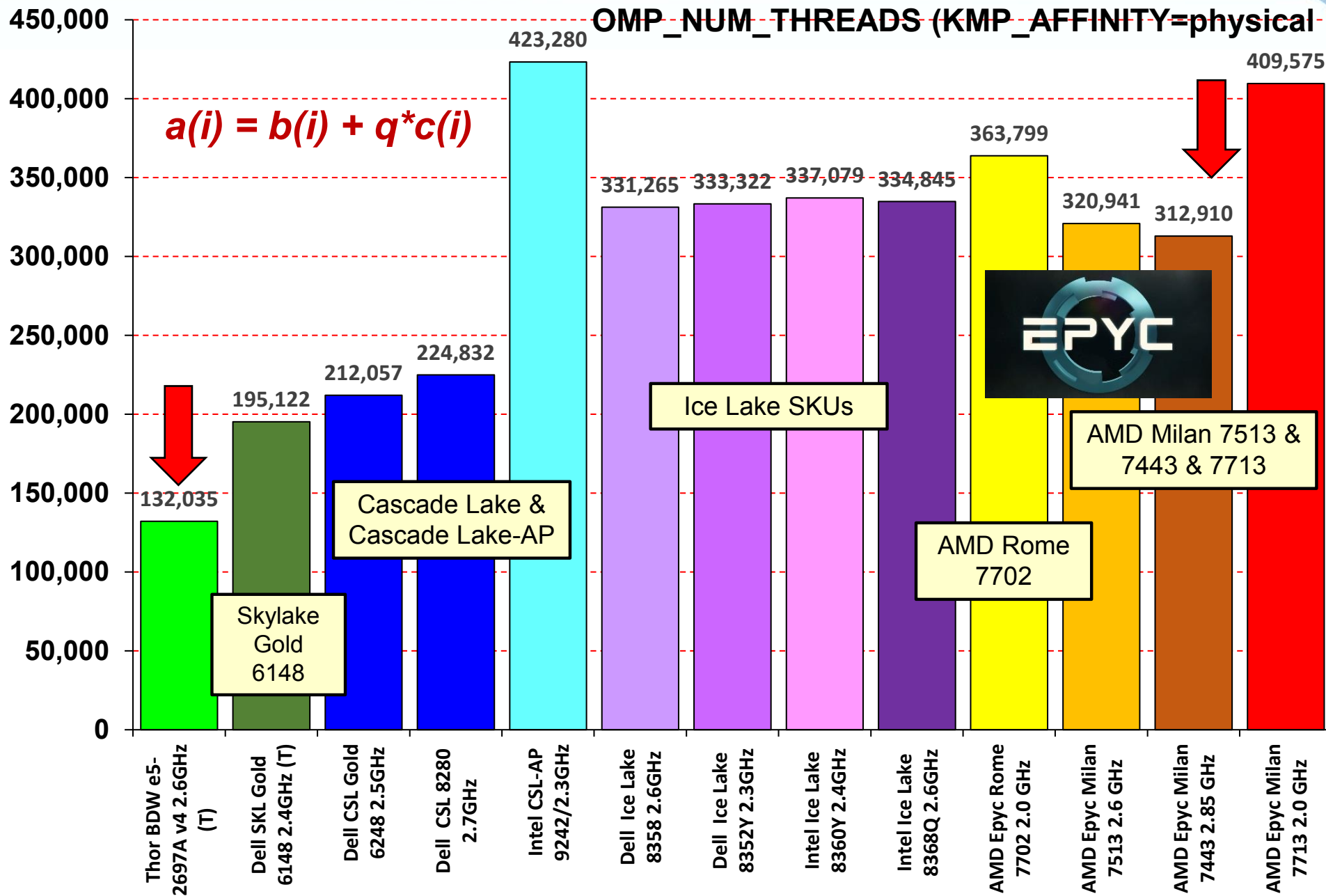
- Martyn Guest for “user benchmarking” with computational chemistry kernels:
 - Uses both “synthetic” and “end-user” benchmarks
 - E.g. stripped down “kernel” or a mix of standard apps
 - Focus on multi-core systems – both CPU and whole system to look at effect of networking etc.
 - Compare job with same number of cores, and also same number of nodes, with different CPUs etc
 - LOT of details in 78 slides in 45 mins ...
<https://epubs.stfc.ac.uk/work/51539351>
 - Also available as a Zoom recording

Memory B/W – STREAM performance

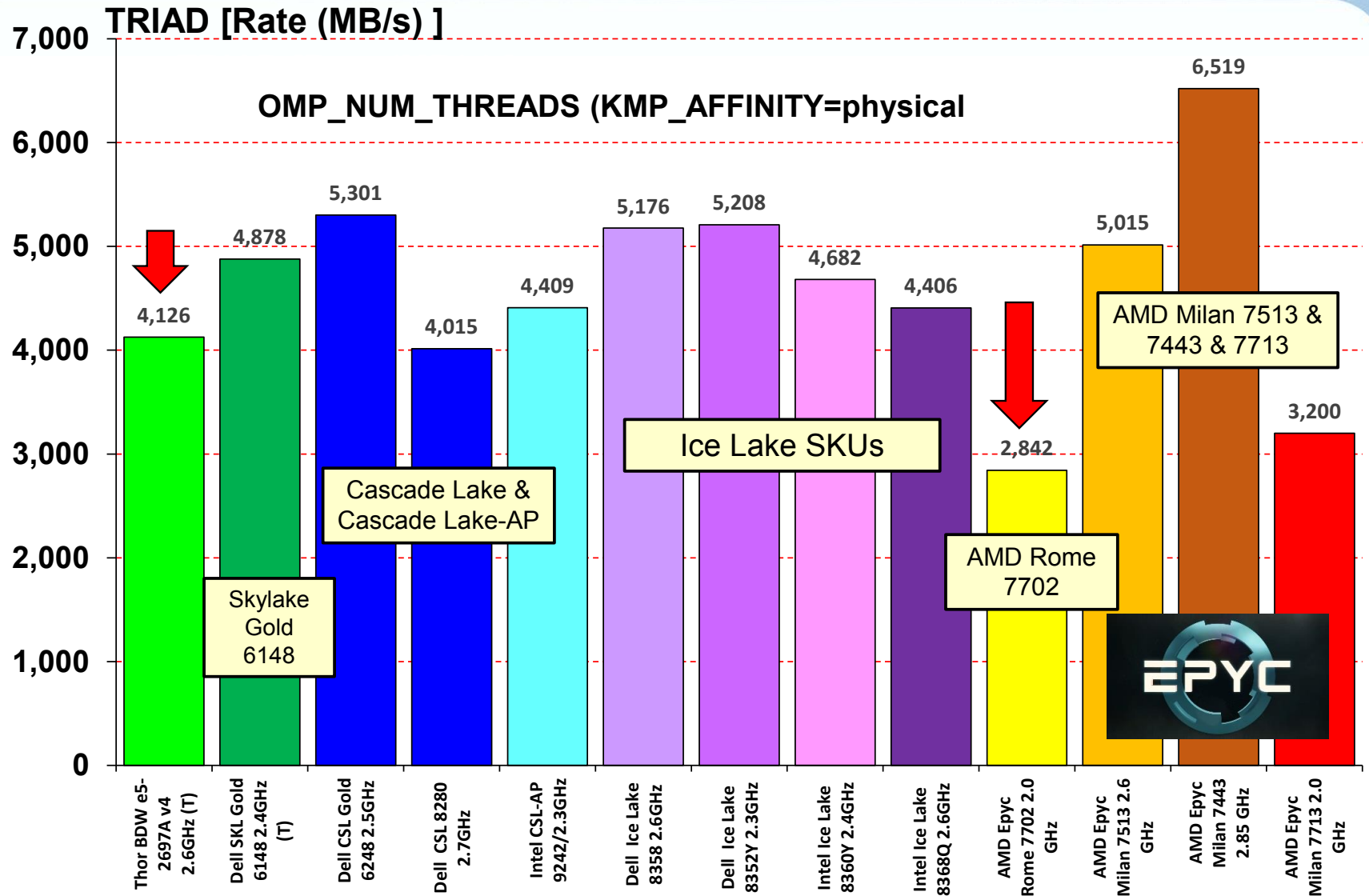
OMP_NUM_THREADS (KMP_AFFINITY=physical

$$a(i) = b(i) + q*c(i)$$

TRIAD [Rate (MB/s)]

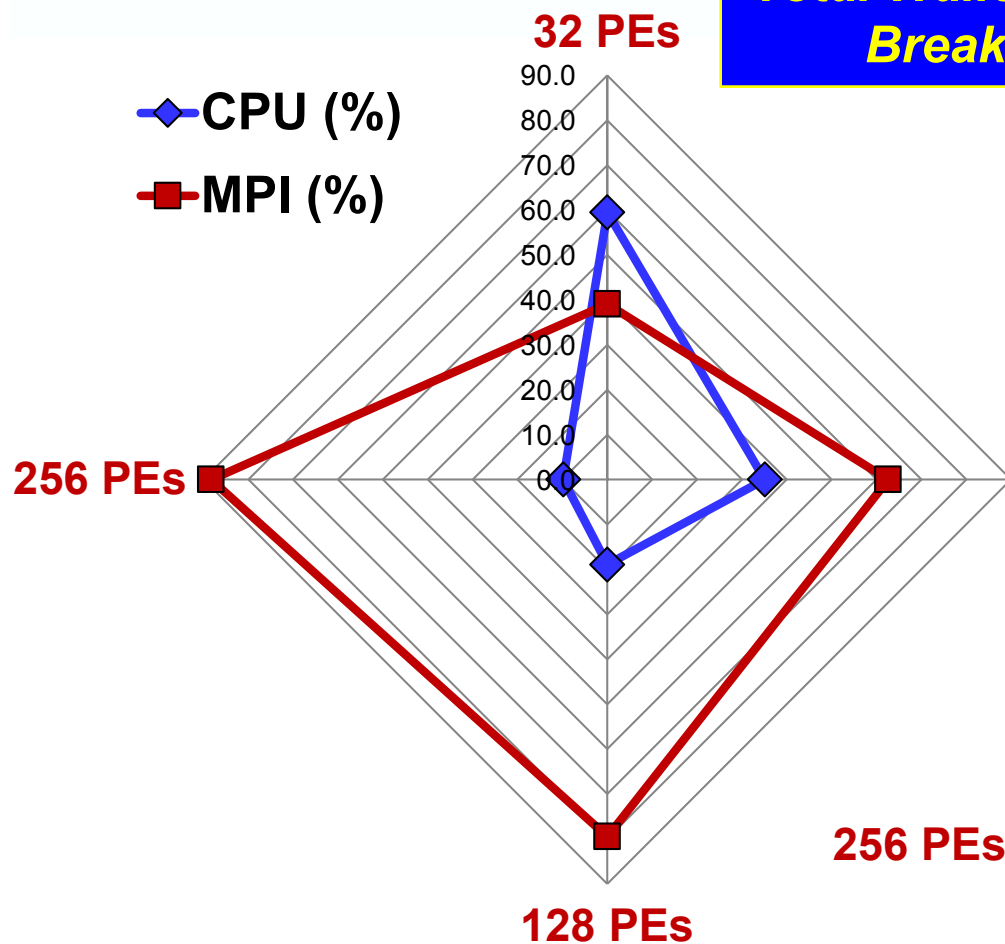


Memory B/W – STREAM / core performance

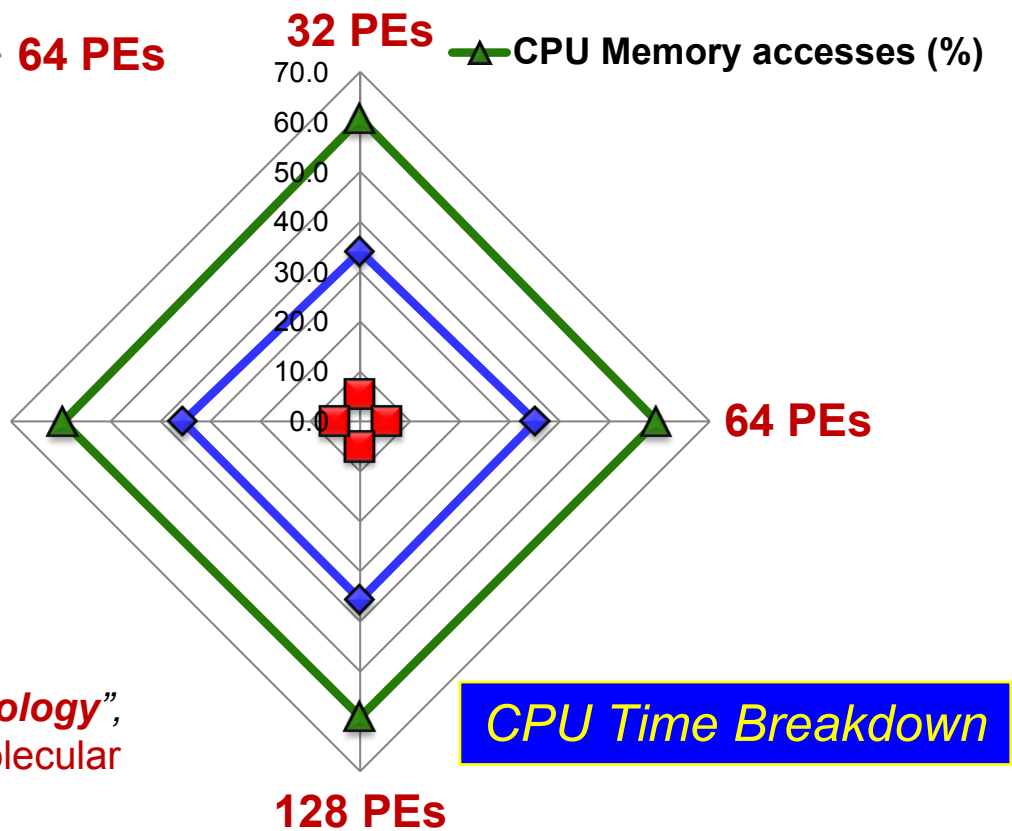


Total Wallclock Time Breakdown

Performance Data (32-256 PEs)



Smooth Particle Mesh Ewald Scheme

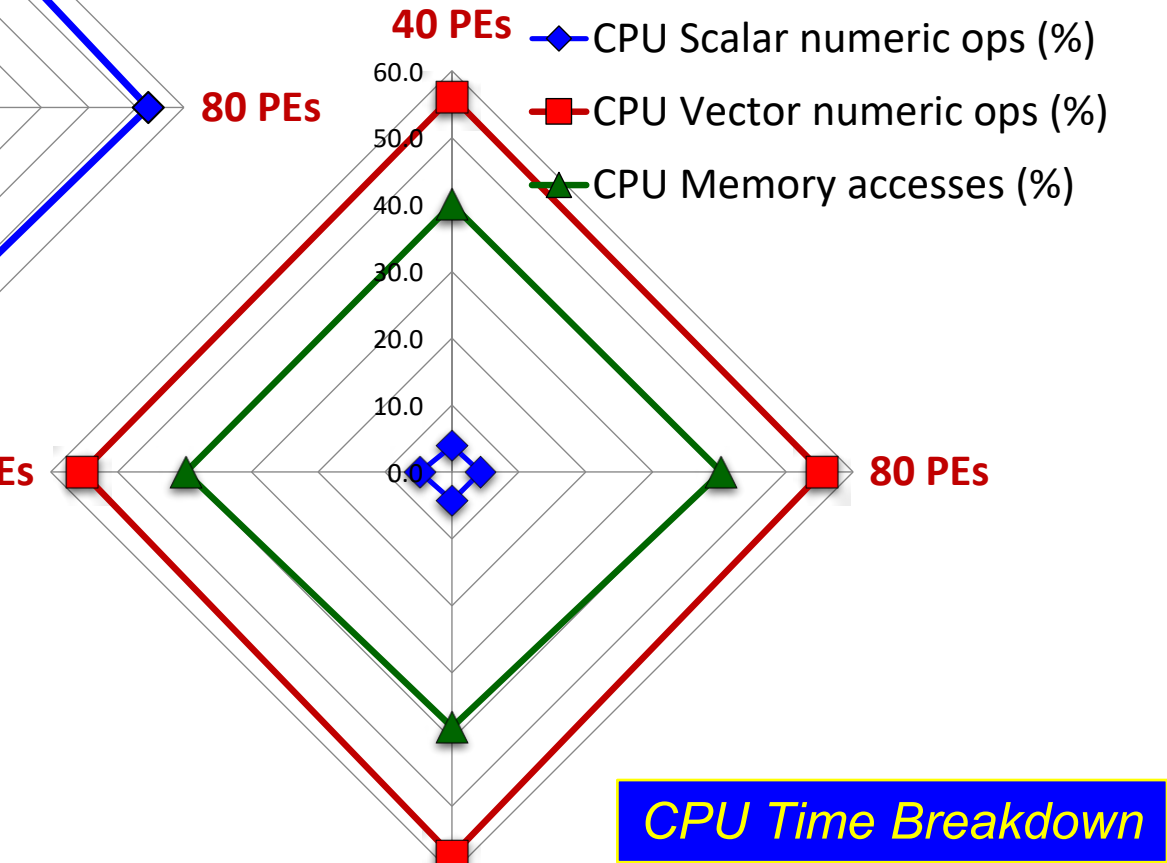
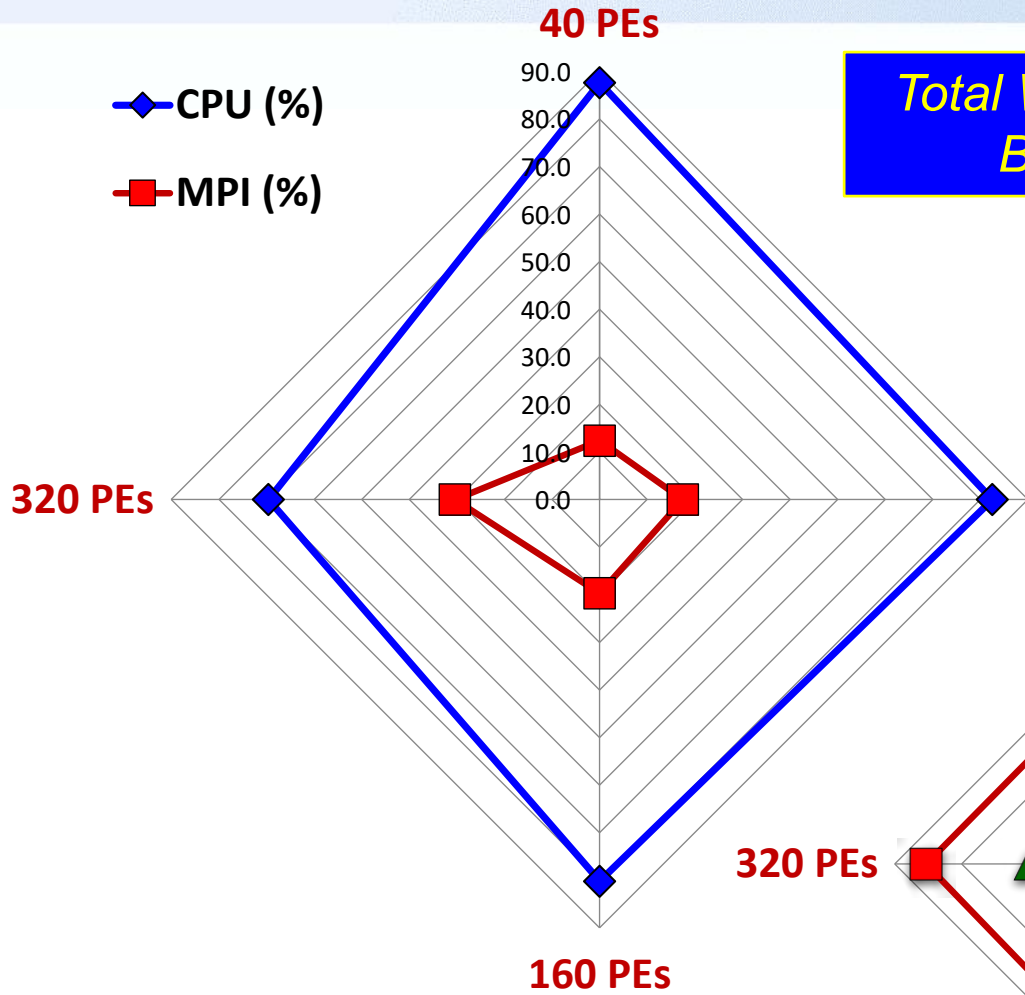


“DL_POLY - A Performance Overview. Analysing, Understanding and Exploiting available HPC Technology”,
Martyn F Guest, Alin M Elena and Aidan B G Chalk, *Molecular Simulation*, (2019) 10.1080/08927022.2019.1603380

CPU Time Breakdown

GROMACS – Ligocellulose Performance Report

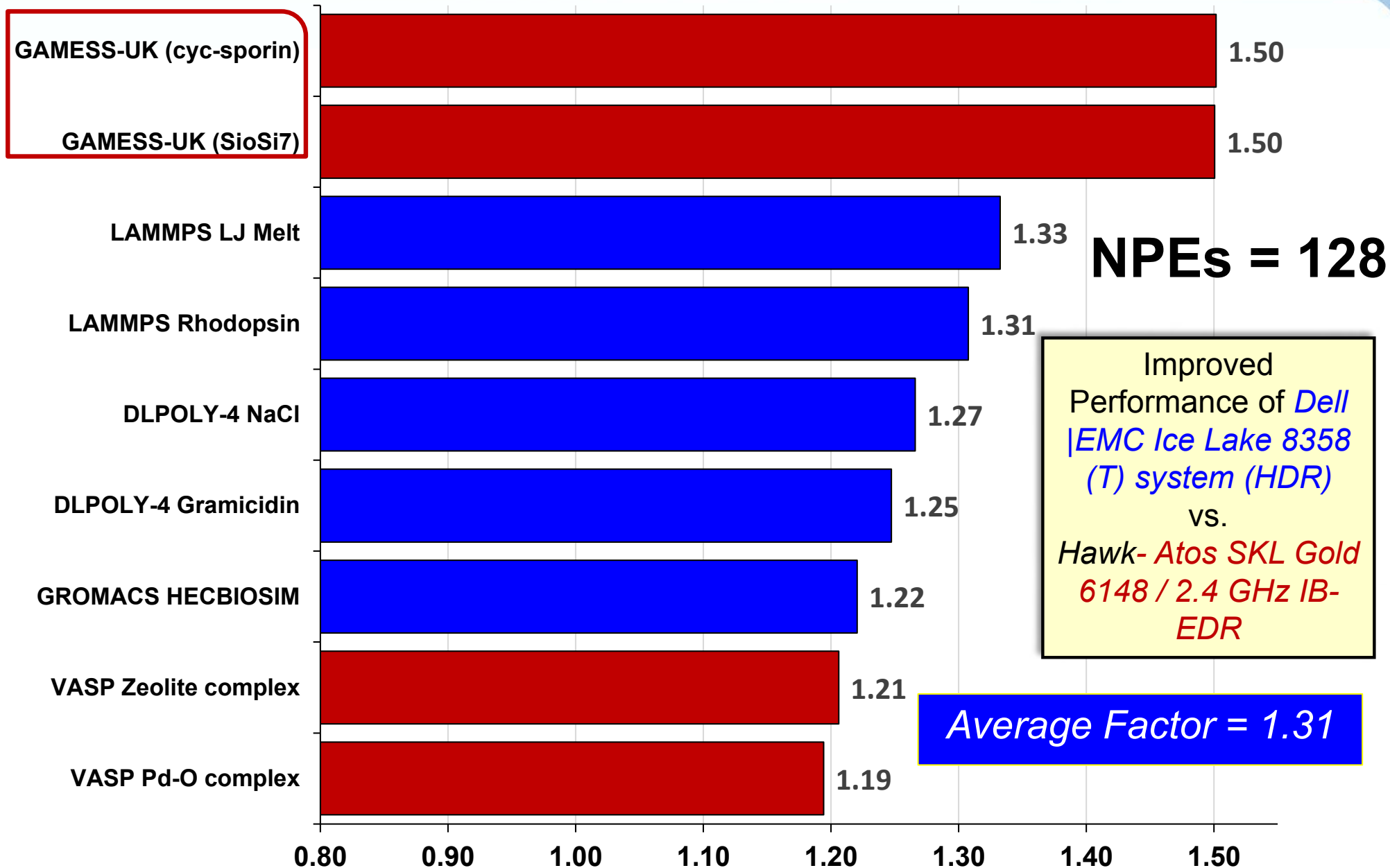
Total Wallclock Time Breakdown



Performance Data (32-256 PEs)

CPU Time Breakdown

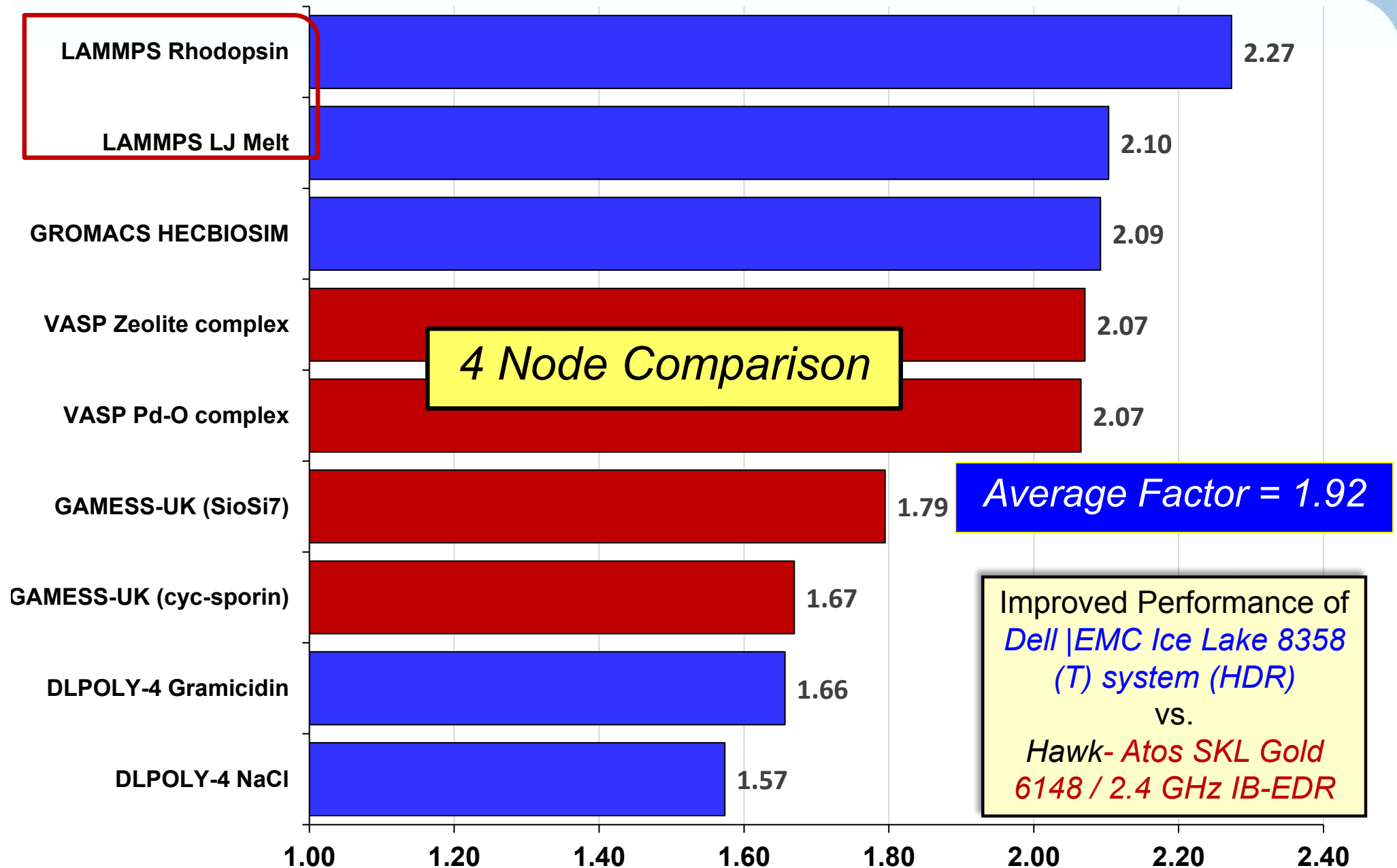
Ice Lake 8358 2.6 GHz HDR vs. SKL 6148 2.4 GHz EDR



Summary – Core-to-Core Comparisons

- A **Core-to-Core comparison** suggests on average that the Intel Ice Lake 8358 2.6 GHz SKU outperforms all others, although relative performance is sensitive to effective use of the AVX instructions.
- Low utilisation of AVX-512 leads to weaker performance of the SKL, CSL and Ice Lake CPUs and **better performance of the Rome & Milan-based clusters** e.g. DLPOLY, LAMMPS
- With significant AVX-512 utilisation, Ice Lake Lake systems outperform the AMD Milan systems in core-to-core comparisons e.g. Gromacs, notwithstanding the use of AVX2-256.
- **Modest improvement** at best on moving from Skylake to Cascade Lake systems – more dramatic improvement moving to Ice Lake.
- **Strong Performance** of the **CSL Gold 6248 system** (2.5Ghz), but surprisingly weak performance from the CSL Platinum 8280 (2.7GHz)
- **Improved CPU** performance of **Spartan 7742 cluster** across all applications compared to CIUK'20 findings (Turbo Mode!)
- Baselined across **P100** and **V100** NVIDIA GPU performance.

Ice Lake 8358 2.6 GHz HDR vs. SKL 6148 2.4 GHz EDR



Summary – Node-to-Node Comparisons

- Given superior core performance, a **Node-to-Node comparison** typical of the performance when running a workload shows the Ice Lake 8358 delivering **superior performance** compared to (i) the SKL Gold 6148 (64 cores vs. 40 cores) by a factor of between 1.6 – 2.3 across all applications.
- The **AMD Rome 7702 and Milan 7713 (128 cores)** along with the **Intel CSX-AP 9242** (96 cores) are the dominant systems given the “high” core counts. e.g., GROMACS and GAMESS-UK.
- The comparison with the SMD Rome-based **Spartan 7742-based cluster** now resolved following the previous reported CIUK performance issues (attributed to Turbo mode)
- **Pricing** – remains of course a key issue, but lies outside the scope of this presentation.

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
 - Hence now our standard approach ...

Further Reading

- Chapter 15 & 16 of “HPC (2nd edition)”, Kevin Dowd and Charles Severance, O'Reilly (1999).
- Top500 at <https://www.top500.org>
- LINPACK at <https://www.top500.org/project/linpack>
- HPC Challenge at <https://icl.cs.utk.edu/hpcc>
- STREAM at <http://sites.utexas.edu/jdm4372/>
- SPEC at <https://www.spec.org>
- SYSmark2018 at <https://bapco.com/products>
- DLPOLY Benchmarking paper – <https://doi.org/10.1080/08927022.2019.1603380>
- Computing Insight UK – see <https://www.scd.stfc.ac.uk/Pages/CIUK2021.aspx>