

Initial Performance Results From HPCx Phase2a

Ian J. Bush

Computational Science and Engineering Department

CCLRC Daresbury Laboratory

Warrington WA4 4AD

I.J.Bush@dl.ac.uk

Acknowledgements

- Mike Ashworth - CCLRC Daresbury
- Martyn Guest - CCLRC Daresbury
- Joachim Hein - EPCC
- Martin Plummer - CCLRC Daresbury
- Fiona Reid - EPCC
- Lorna Smith - EPCC
- Andy Sunderland - CCLRC Daresbury

- And all the other members of the terascaling team at DL and EPCC

In 2004 the HPCx was upgraded (Phase2)

- 1.7 GHz p690+ frames instead to 1.3GHz p690
- Upgrade to IBM's High Performance switch (HPS)

Very recently we have had another upgrade

- 1.5 GHz p5-575 frames
- 2 Gbytes per processor instead of one

HPCx - Phase 1

- Collaboration between CCLRC DL, University of Edinburgh and IBM
- Service started December 2002
- 1280 IBM 1.3GHz Power 4 processors
- 1 GByte perprocessor
- 32 Way SMPs
- Colony switch
- 3.24 Tflops sustained on Linpack



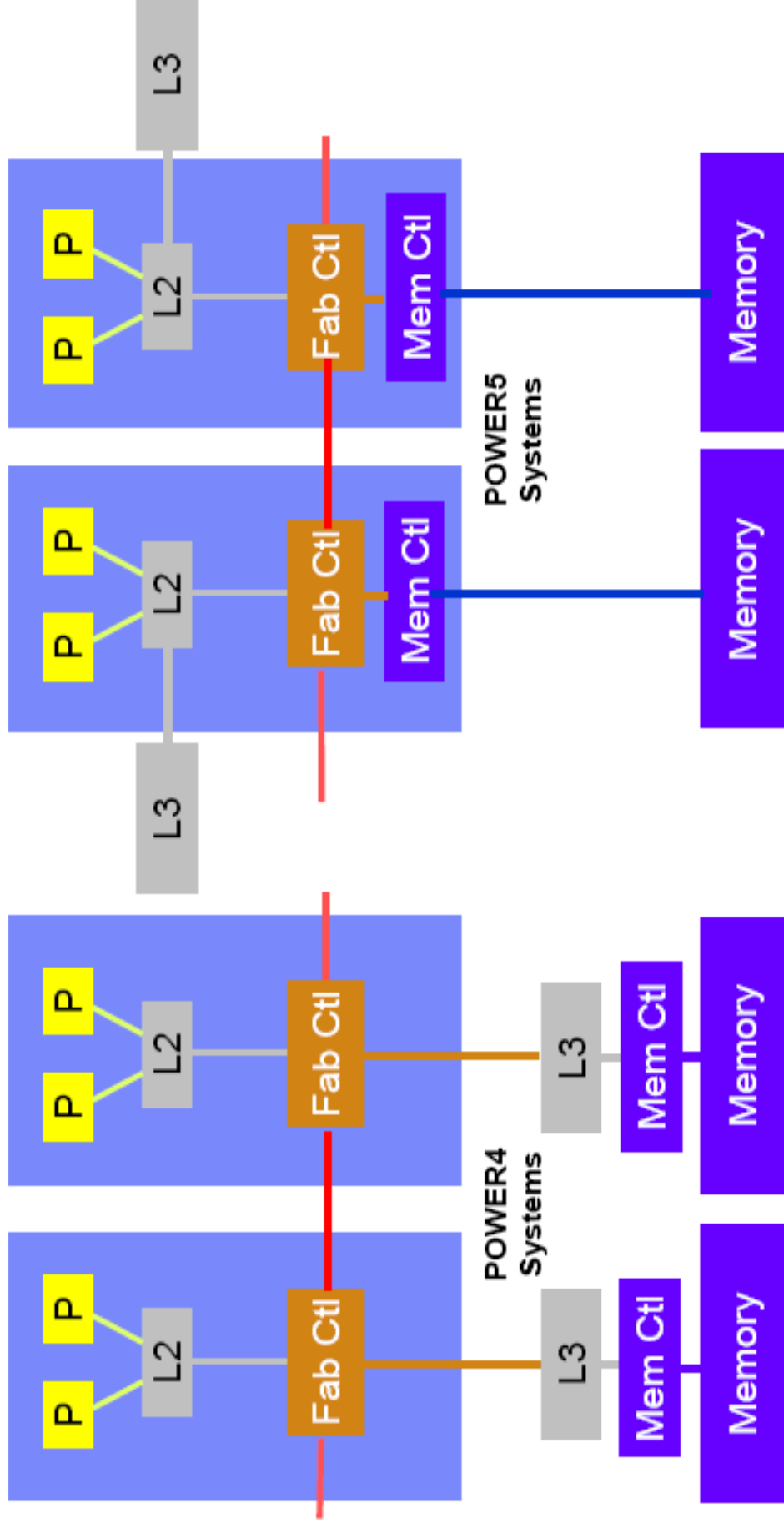
- Service started April 2004
- 1600 IBM 1.7GHz Power 4+ processors
- 1 Gbyte per processor
- 32 way SMPs
- HPS interconnect
 - SP7 installed July
 - SP9 installed October
- 6.188 Tflops sustained on Linpack

???

- Service started November 2005
- 1536 IBM Power 5 processors
- 2 GBytes per processor
- HPS interconnect
- 7.395 Tflops sustained on Linpack
- 16 Way SMPs
- Much smaller physical footprint

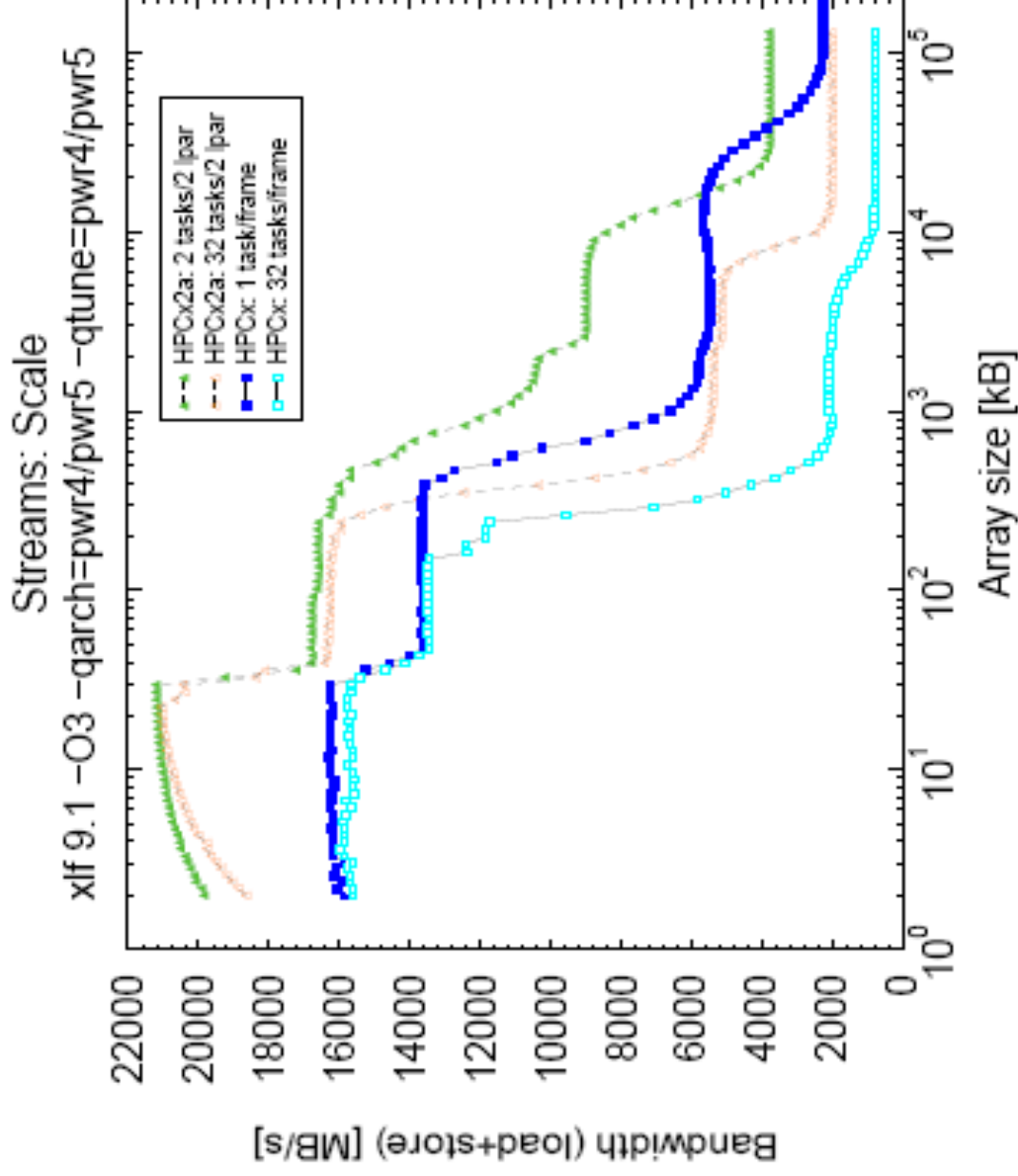
???

POWER4 System Evolution to POWER5 Systems



- Perhaps the most important is the memory upgrade
 - About twice as much memory on the machine
 - But I won't say much about that today
- Despite the smaller number of processors and the slower clock the Linpack rating is almost 20% faster!
 - Alternatively the new processors can get a much higher percentage of peak
 - So we expect codes that can exploit the cache to run better
 - Also helped by L2 and L3 being slightly bigger
 - Stream benchmark shows cache improvement

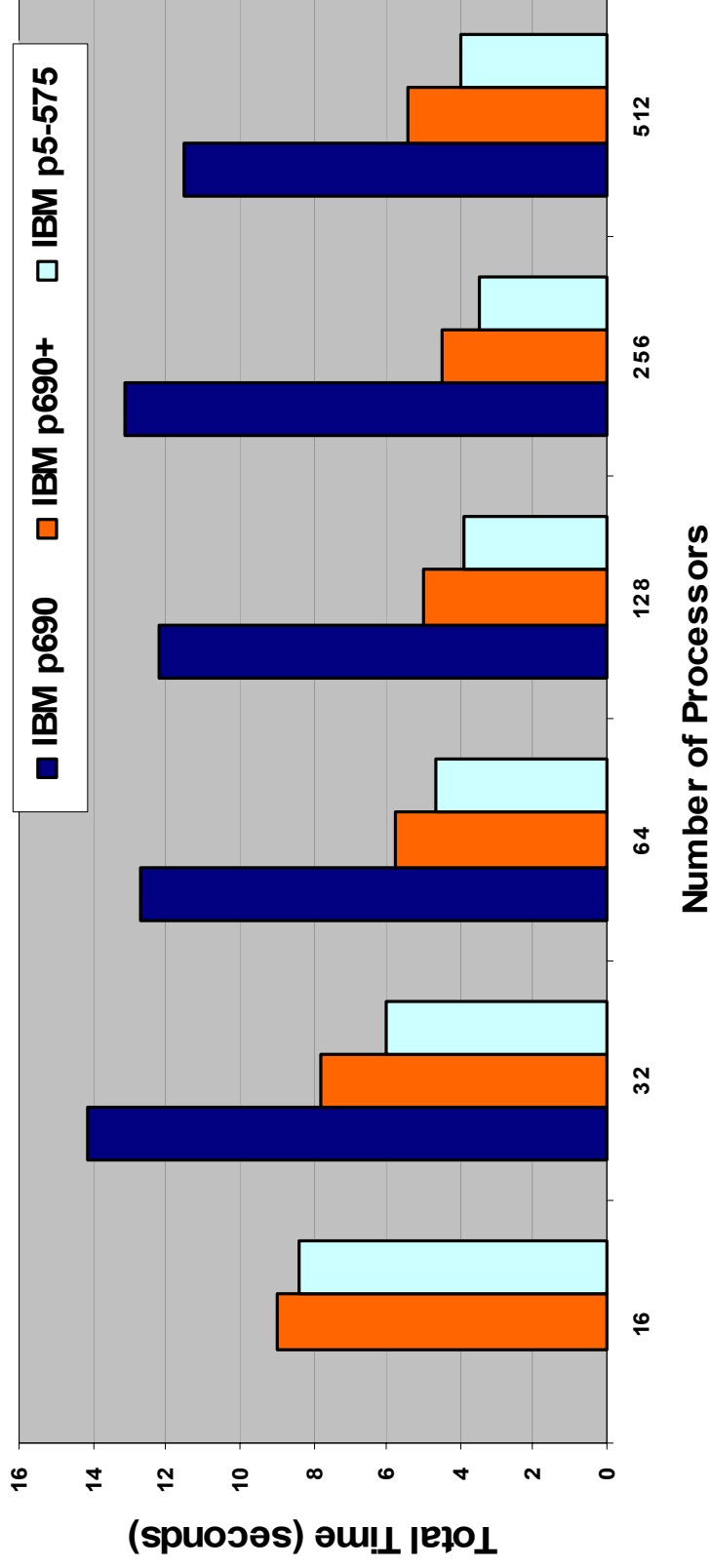
Comparison of the Two Systems



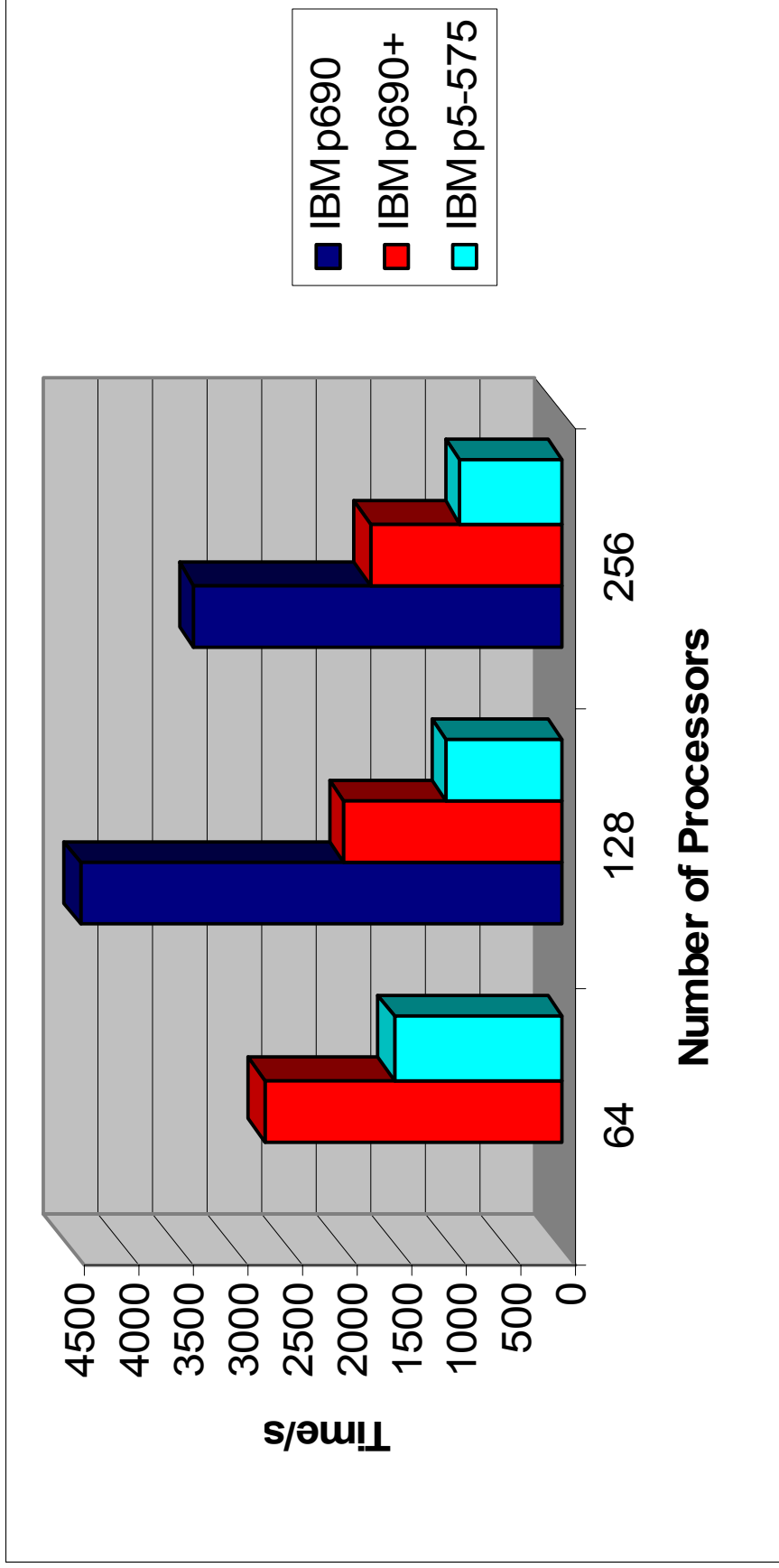
- In general real applications that are dominated by operations that benefit from the improved cache architecture show a marked improvement on Power5
 - CASTEP
 - CPMD
 - H2MOL
 - AIMPRO
 - ...

But let's start with a kernel just to check that the LINPACK story carries over

PDSYEVD Performance for Fock Matrix, (CRYSTAL), N_basis = 3888

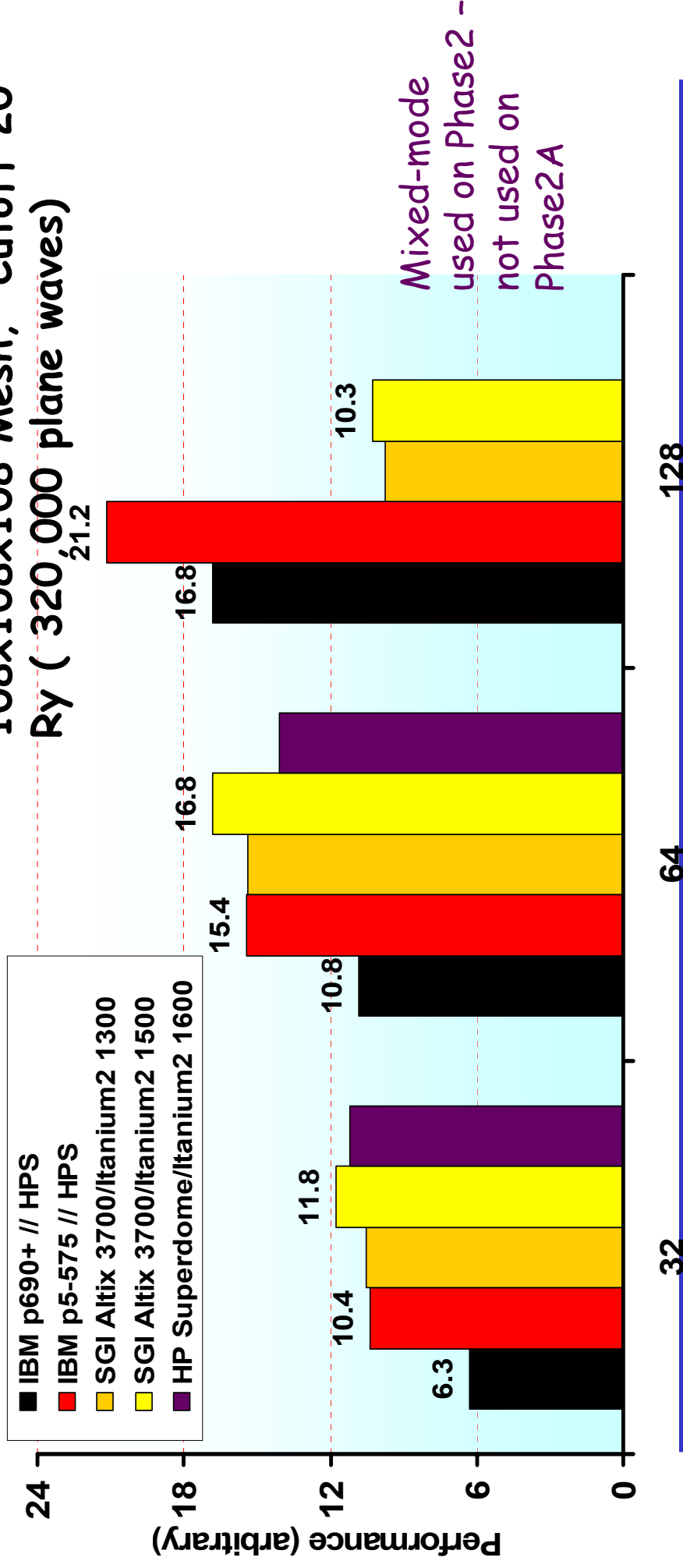


- CASTEP and CPMD are *ab initio* electronic structure codes that use a plane wave basis
- In both the most important operation is multidimensional FFTs
 - Can be very well optimized for caches
- Less important but still significant are a number of matrix matrix operations



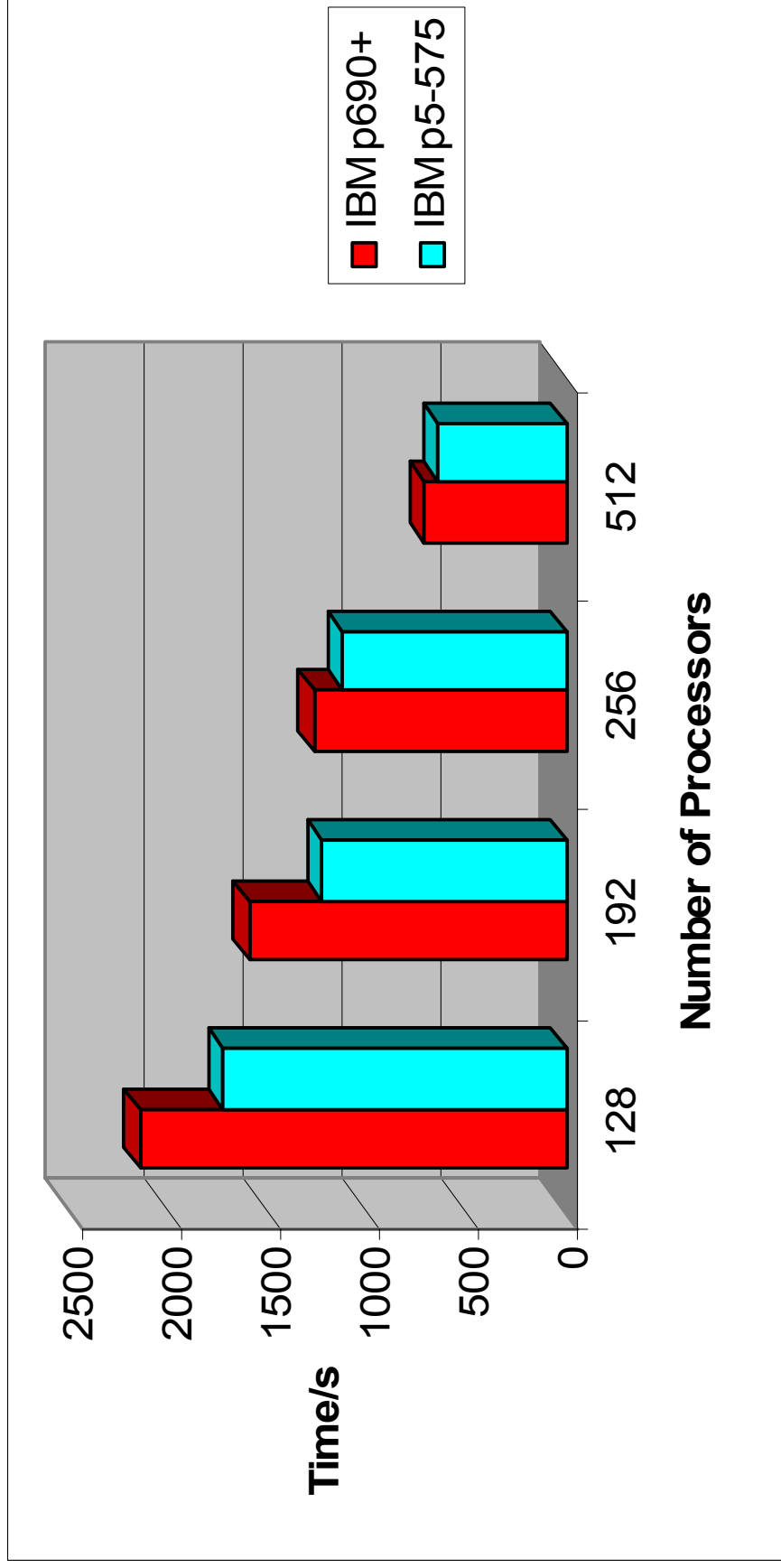
Developed at IBM Zurich from the original Car-Parrinello Code in 1993 (www.cpmid.org); Hutter & Curioni

512 atoms Si cluster,
108x108x108 Mesh, Cutoff 20
Ry (320,000 plane waves)



- We have only run H2MOL on one processor count
- On 496 Processors it is 1.54 times faster on Phase2a as opposed to Phase 2
- Dominated by ZGEMM operations

- AIMPRO is another electronic structure application. This time, however, a gaussian type basis is used.
- Matrix matrix operations and eigensolvers are very important in this code.



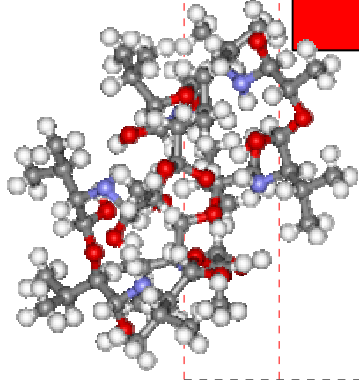
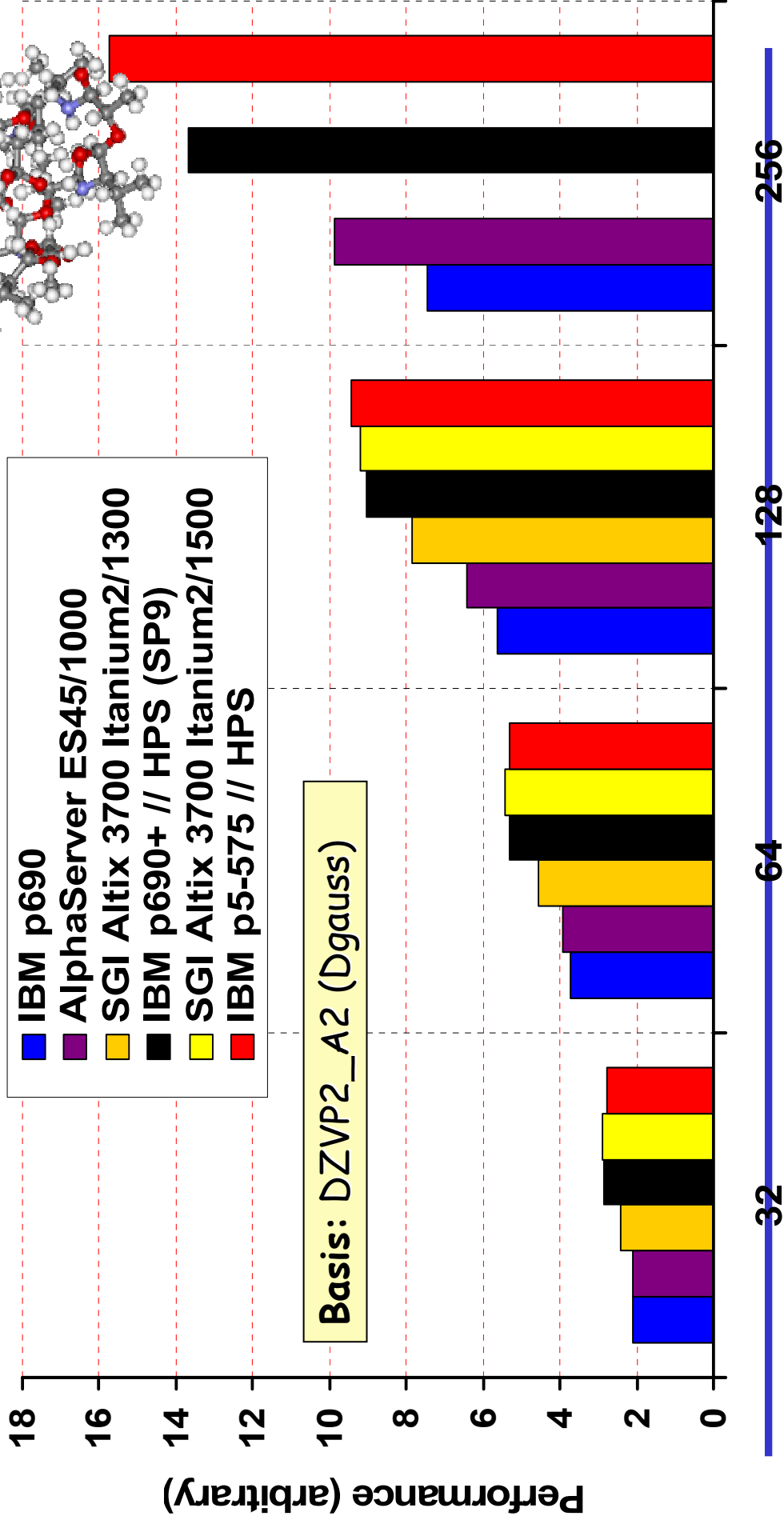
Improved Cache - Better Scaling !

- In some codes the improvement in the cache architecture results in improved scaling
- This can occur if the time taken in a poorly scaling part of the algorithm is reduced compared to the other (better scaling) parts
- GAMESS-UK is an example of this
 - The eigensolver scales poorly while the integrals scale almost perfectly

Performance of the GA-based Implementation

Valinomycin, 1620 GTOs

DFT HCTH: IBM p5-575 and High-end Systems

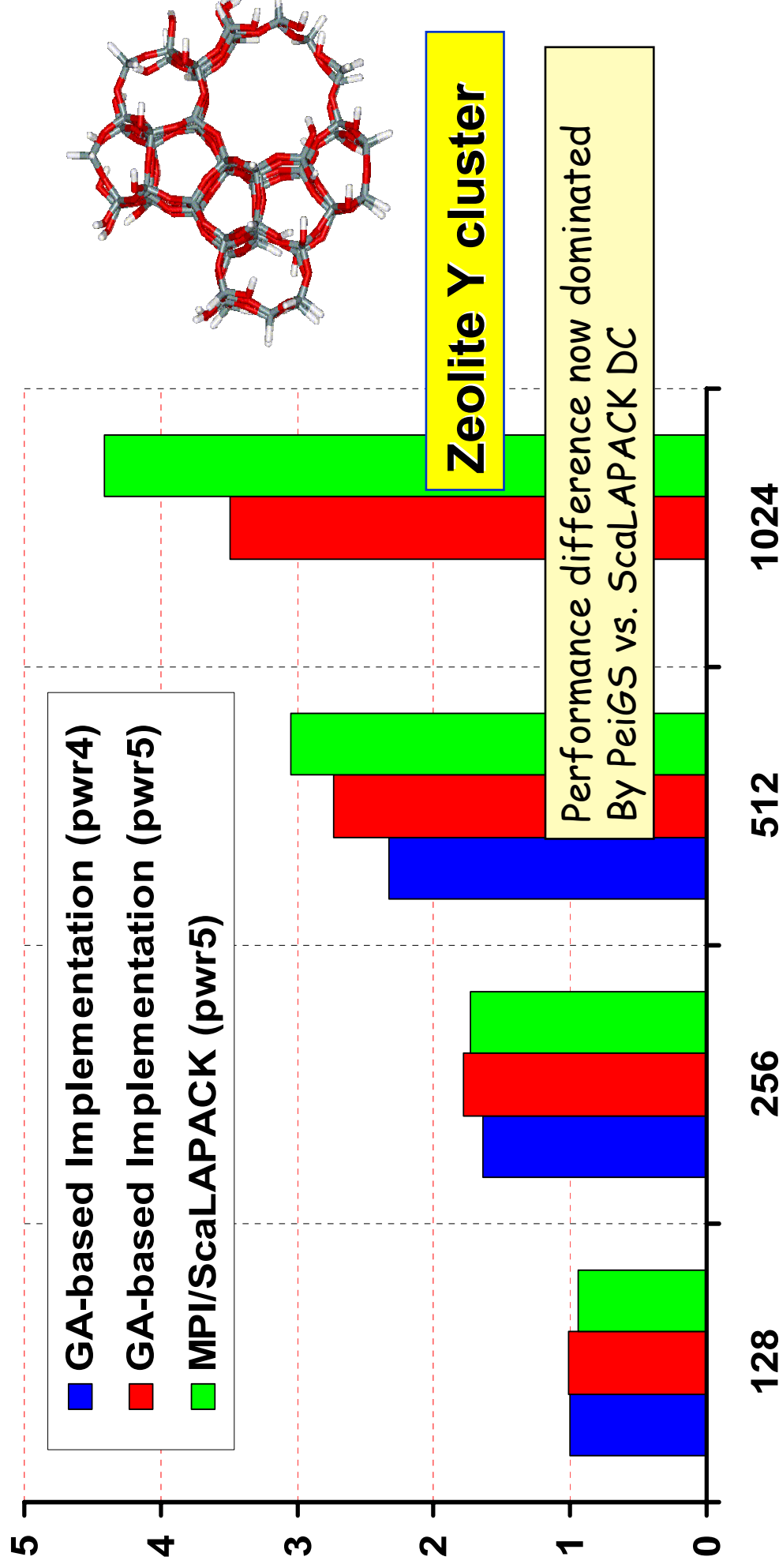


GAMESS-UK

DZVP Basis (DZV_A2) : 3975 GTOs

Hartree Fock (IBM p690+ and p5-575)

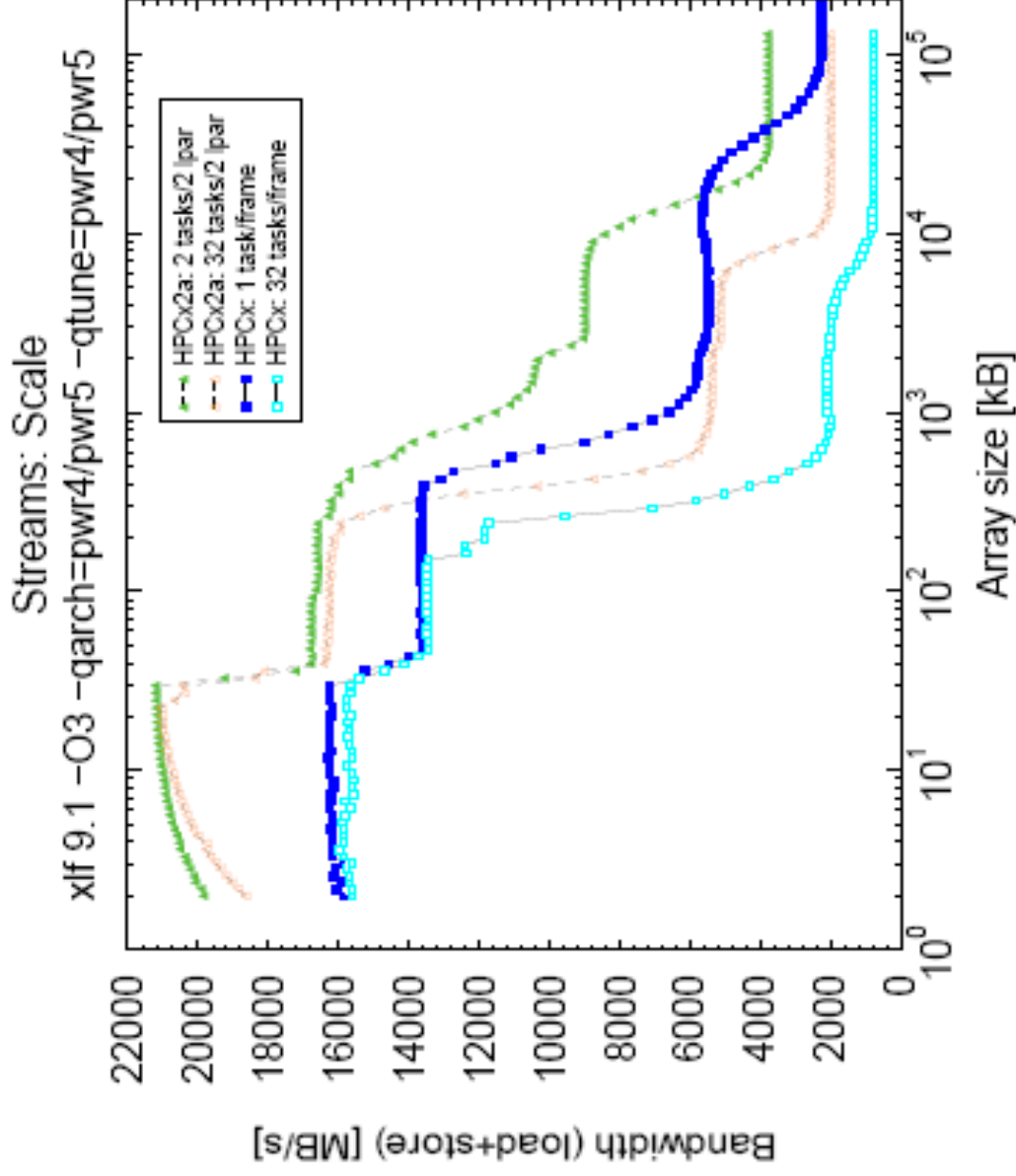
Performance



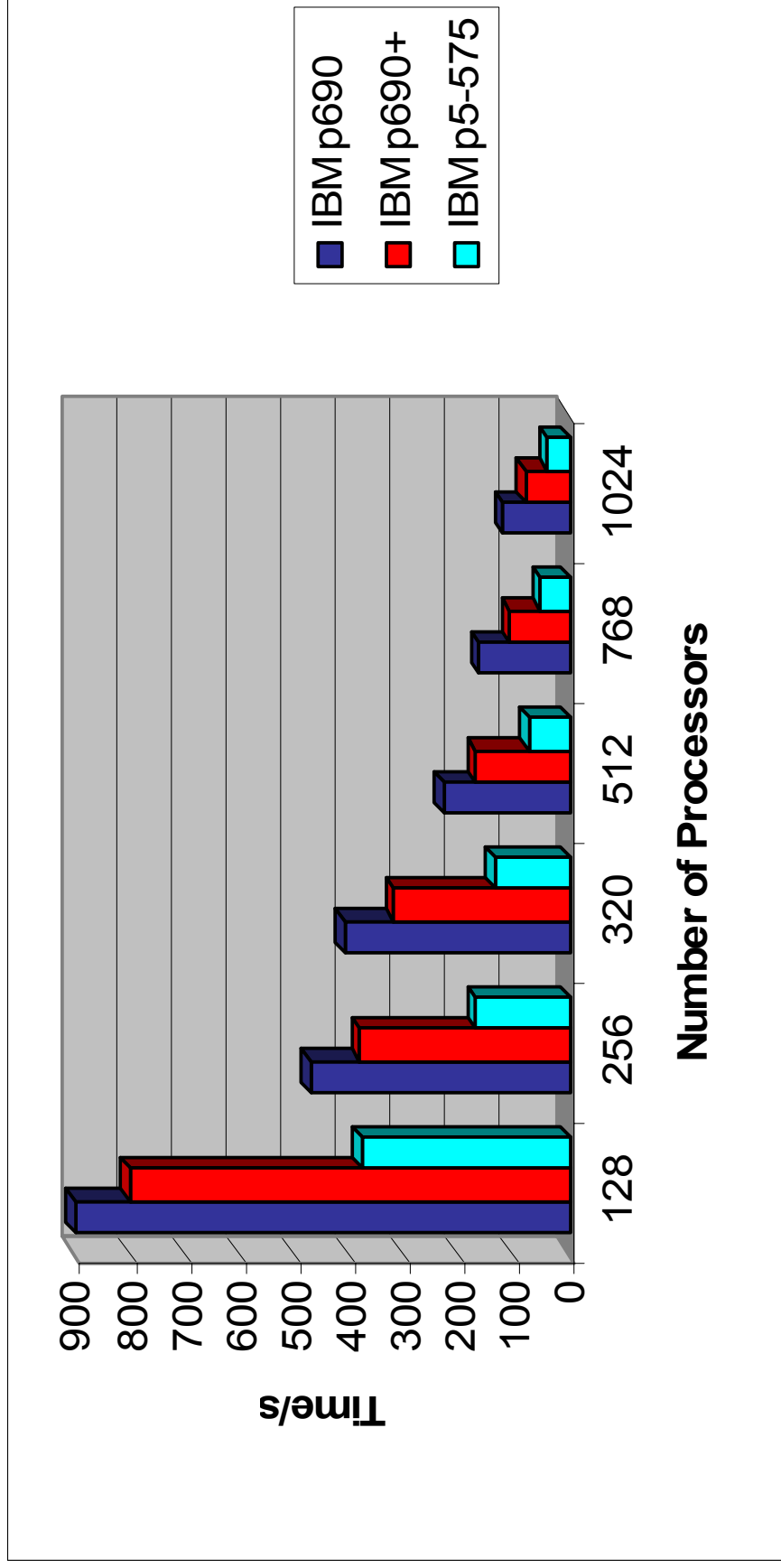
But I'm not in Cache !

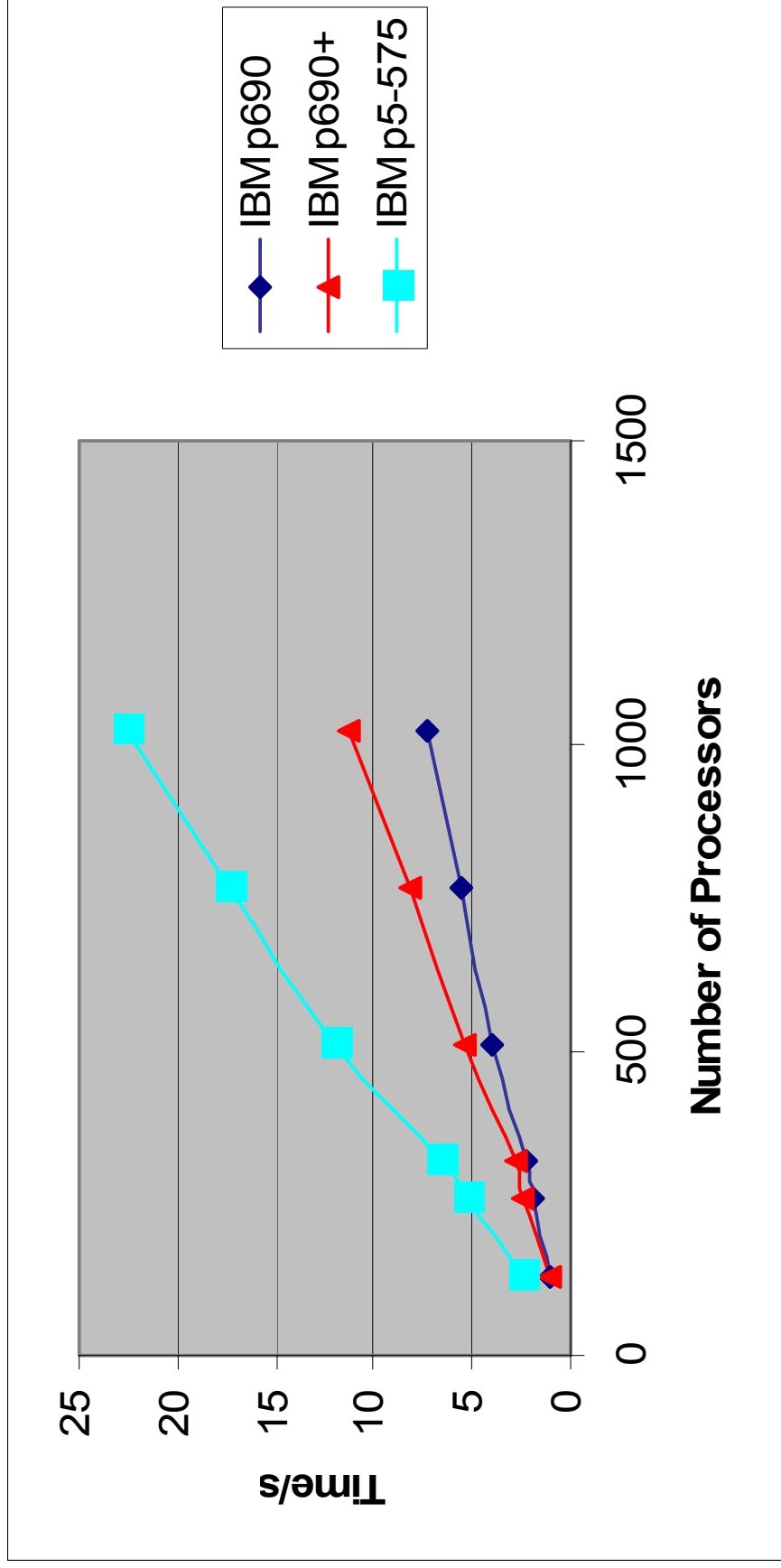
- Another look at the Streams benchmark is useful [here](#)

Comparison of the Two Systems



- As well as the bandwidth to cache the bandwidth to main memory has more than doubled on phase2a
- Should massively improve the performance of codes that do not re-use data

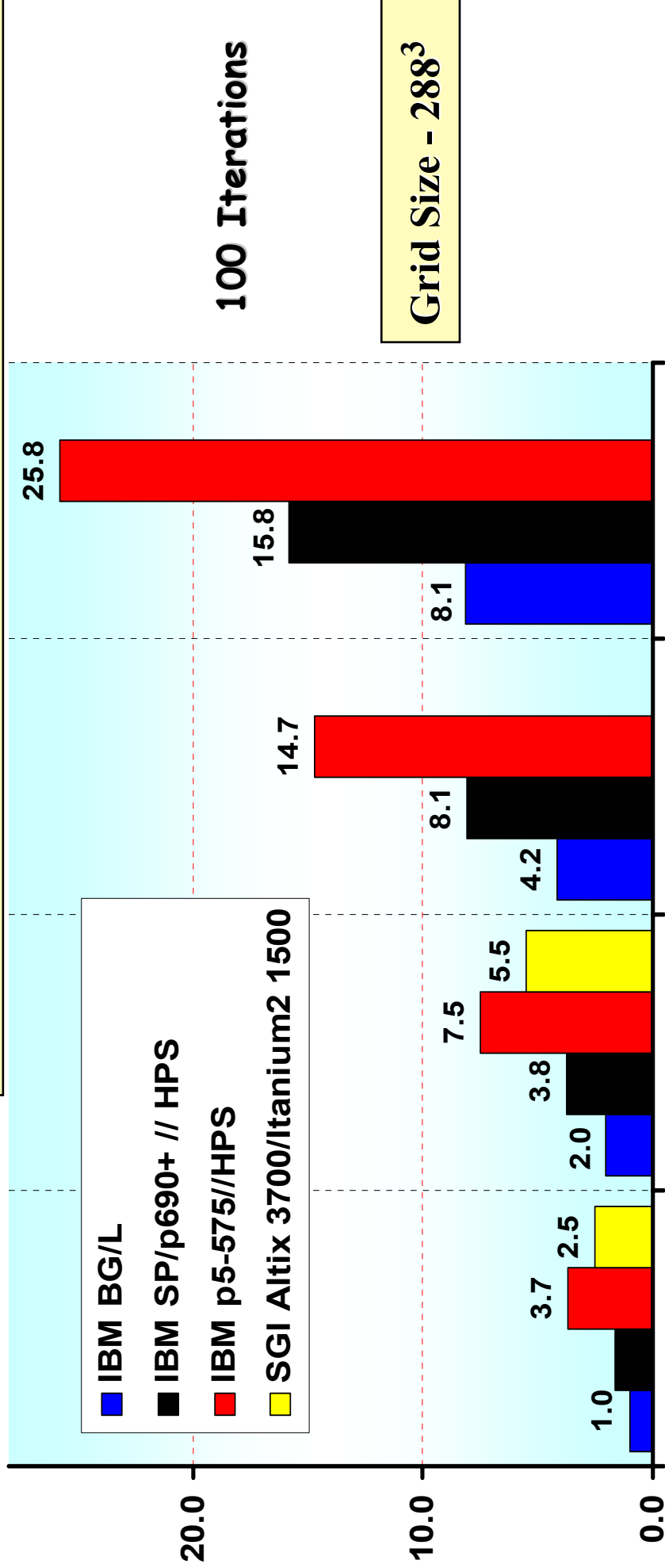




Direct numerical simulations (DNS) of turbulent pre-mixed combustion solving the augmented Navier-Stokes equations for fluid flow. Pressure solver utilises either a conjugate gradient method with modified incomplete LU pre-conditioner or a multi-grid solver (both make extensive use of Level 1 BLAS) or fast Fourier transform.

Conjugate Gradient + ILU

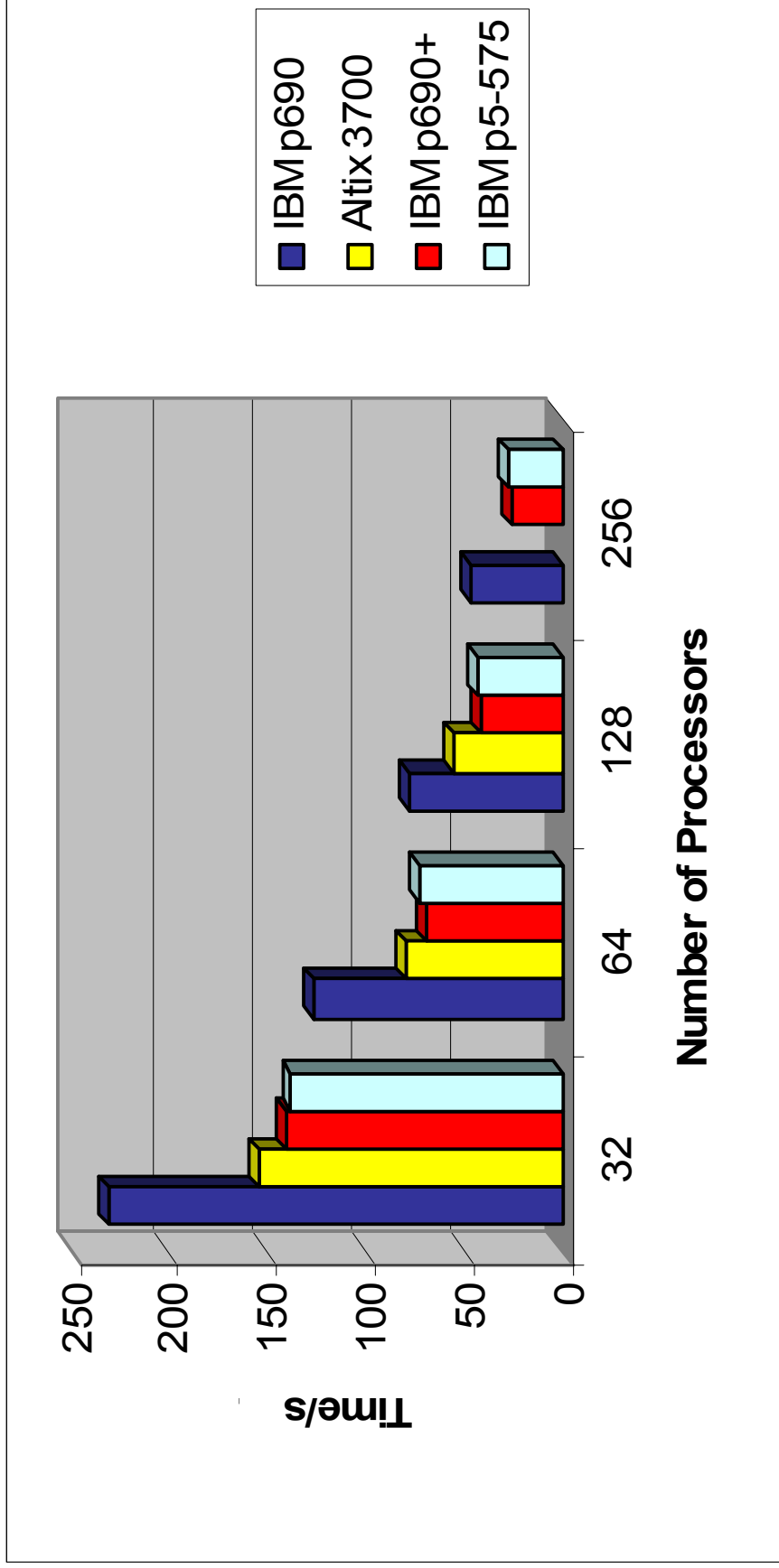
Performance



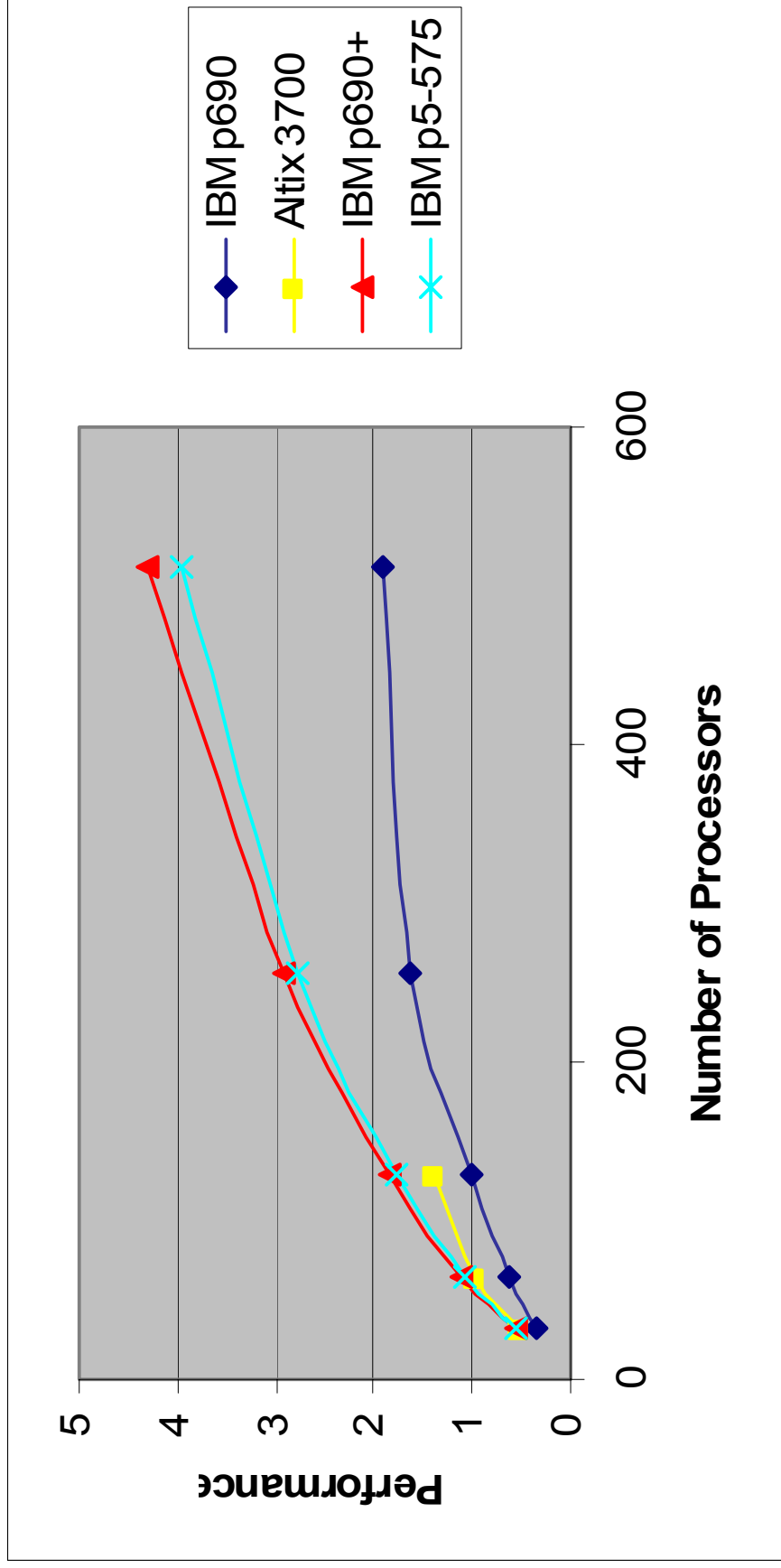
So is it all good news ?

- One class of commonly used applications on HPCx at best show a very small improvement on Phase2a
 - Often show a performance degradation
- That set of applications is classical Molecular Dynamics

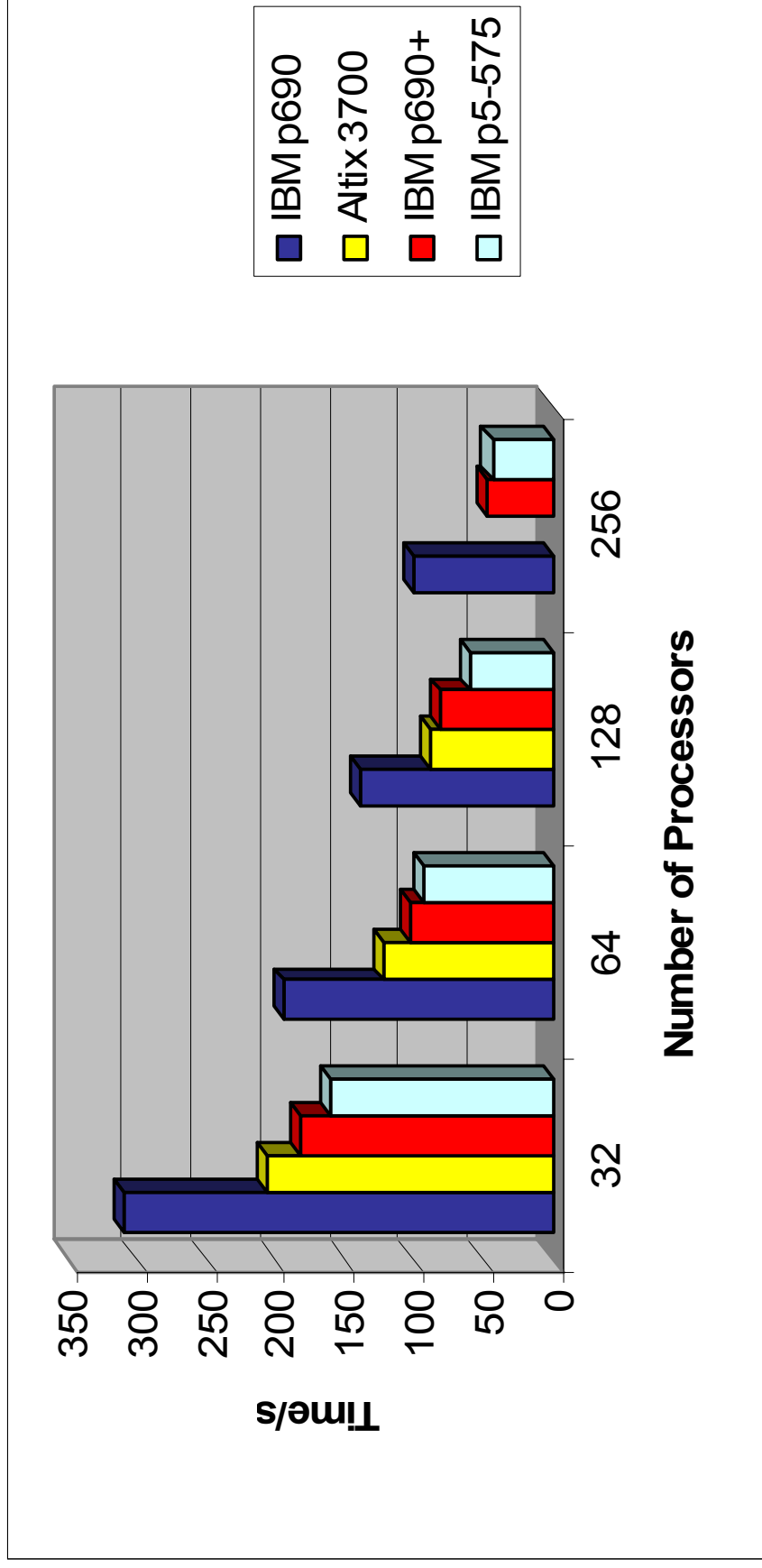
DL_POLY 3 - NaCl



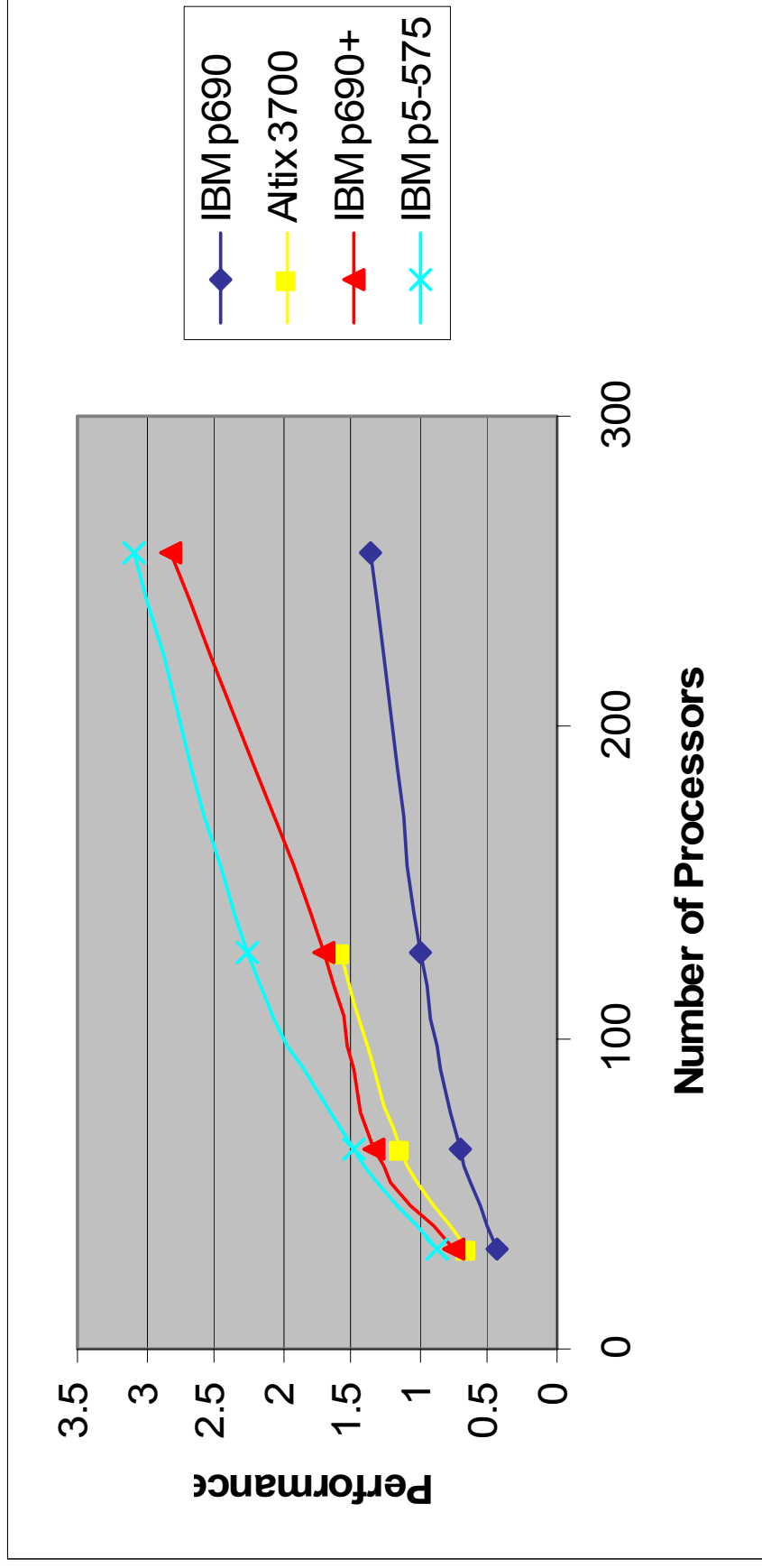
DL_POLY 3 - NaCl



DL_POLY 3 - Gramicidin

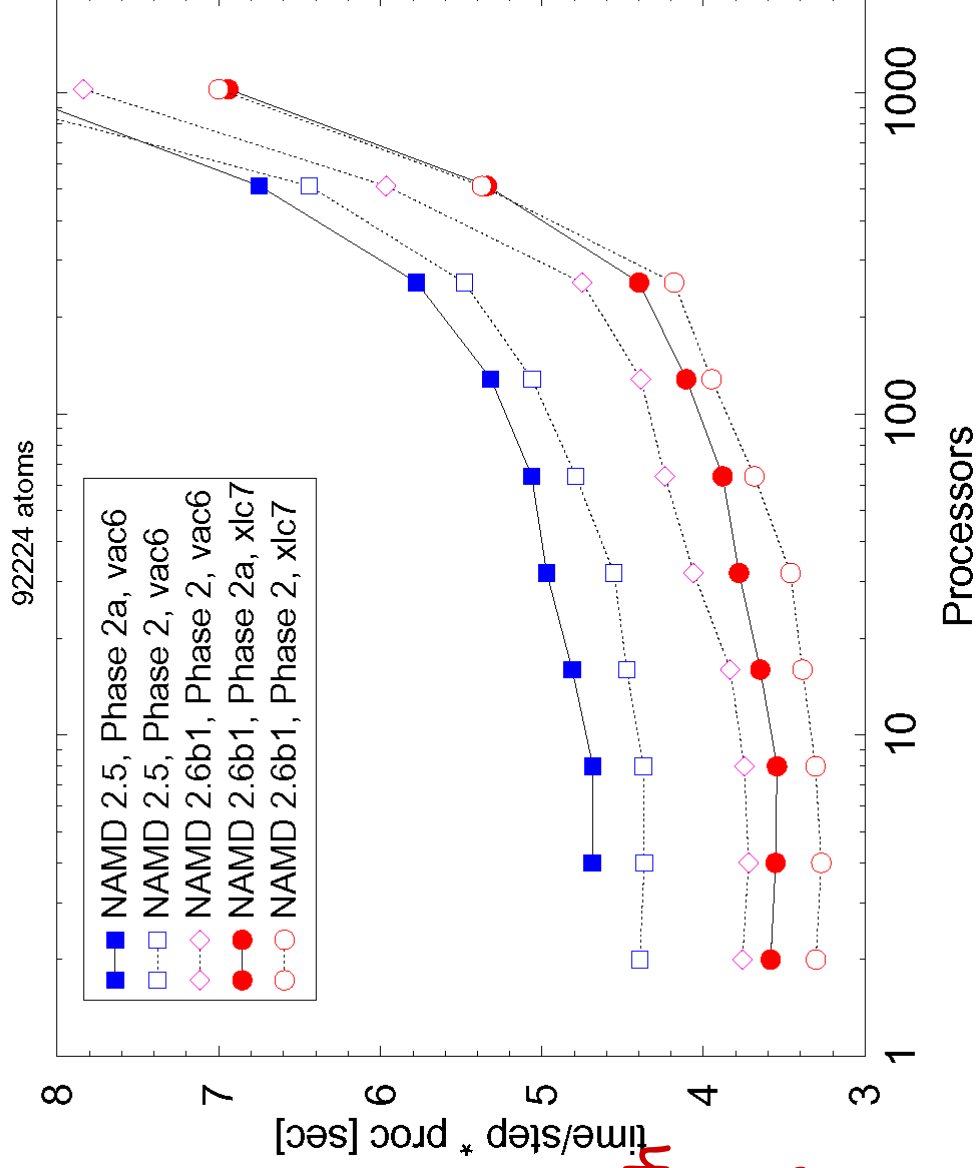


DL_POLY 3 - Gramicidin

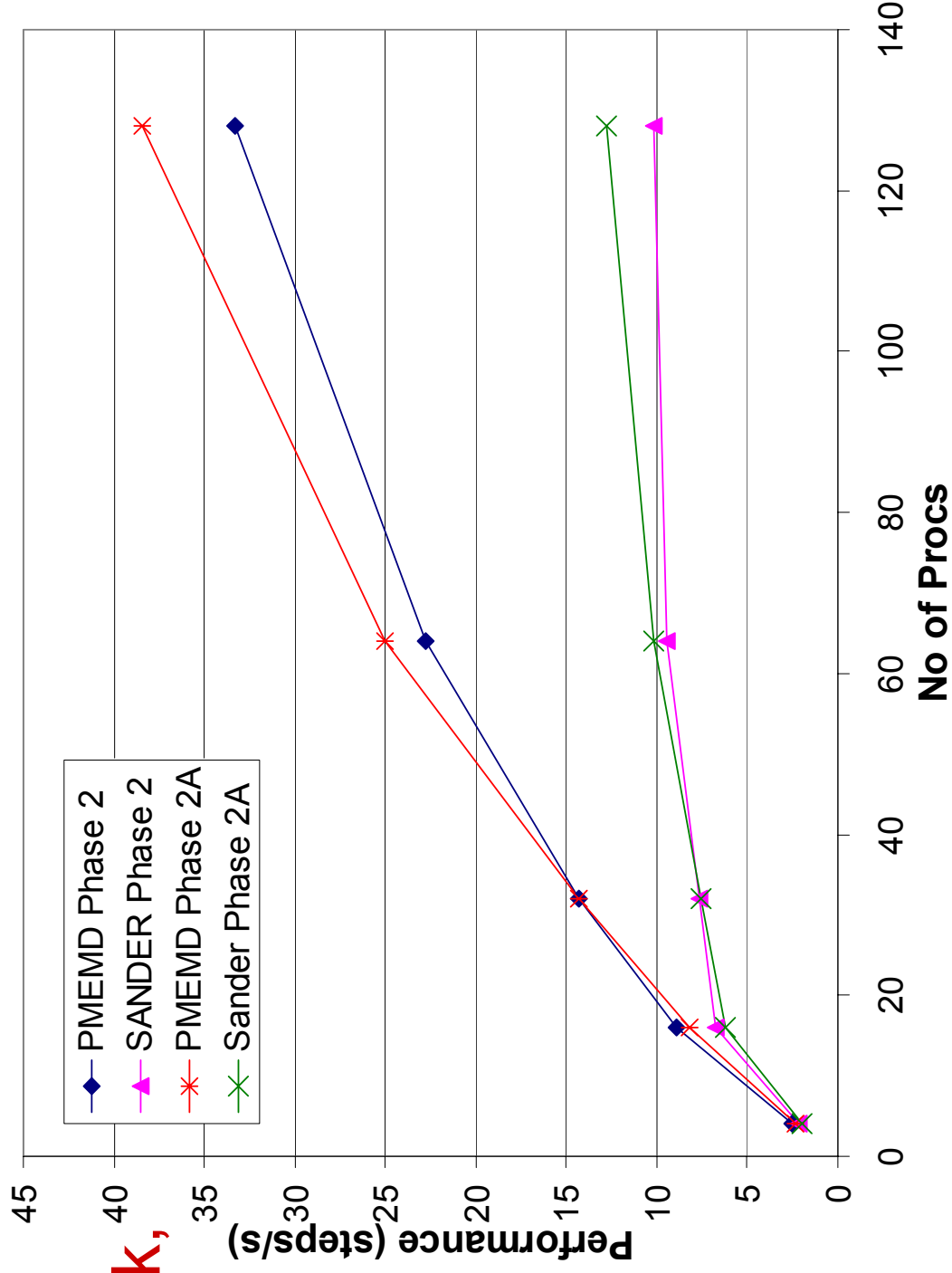


- Results on Phase 2a: **preliminary!!!**
- Substantial performance boost from NAMD 2.5 to 2.6b1
- New compiler helps NAMD 2.6b1
- Phase 2a slightly slower, but not as much as clock
- Phase 2a scales better

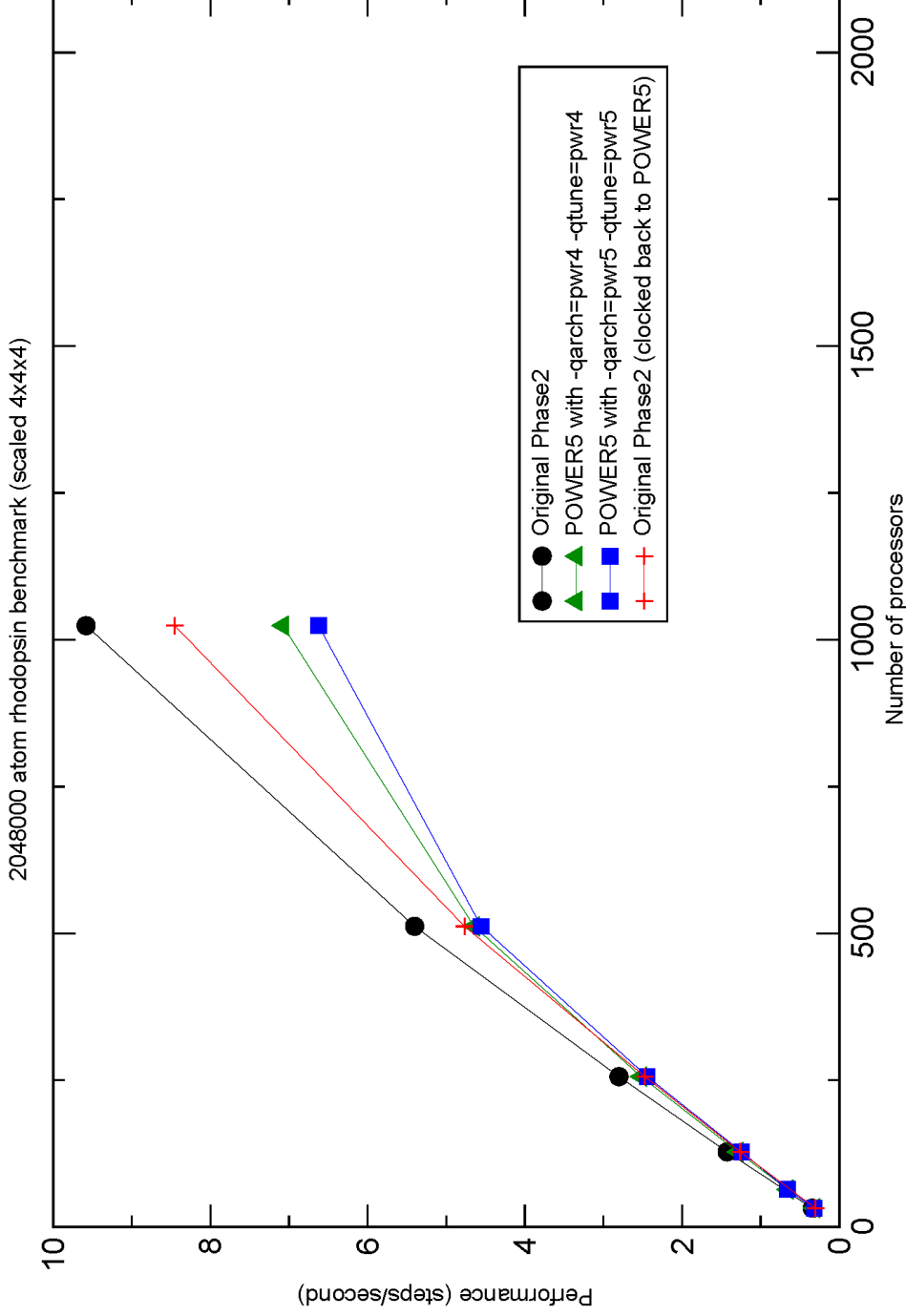
NAMD APO-A1 benchmark



- IX protein benchmark, 90906 atoms



LAMMPS 2005, HPCx Phase2/POWER5 performance comparison



2 million atom
Rhodopsin
benchmark with
negligible I/O

Up to 512
procs performs
with clock ratio

Beyond 1024
processors
POWER5 scales
less well - now
investigating

- Classical MD codes spend a lot of time looking through tables to decide which set of atoms next to consider
 - Often strided accesses in memory result
 - Application can become sensitive to latencies in any part of the memory subsystem
- Unfortunately I do not yet have the numbers that compare memory latencies in the Power5 system with those in the Power4+

- The memory subsystem architecture has improved markedly in Power5
 - The higher memory bandwidth to cache results in a higher percentage of peak performance being obtainable
 - Slower clock yet higher performance!
 - The higher bandwidth to main memory results in applications that do not reuse data (much) but do have stride one access patterns run much faster
- HOWEVER initial results suggest that memory latency has not improved as much
 - This needs more investigation
- **BUT YOU'VE GOT TWICE THE MEMORY**