Parallel 3D-FFTs for multi-core nodes on a mesh communication network

Joachim Hein[1,2], Heike Jagode[3,4],
Ulrich Sigrist[2], Alan Simpson[1,2],
Arthur Trew[1,2]

[1]HPCX Consortium
[2]EPCC, The University of Edinburgh
[3]The University of Tennessee in Knoxville
[4]Oak Ridge National Laboratory (ORNL)

Outline

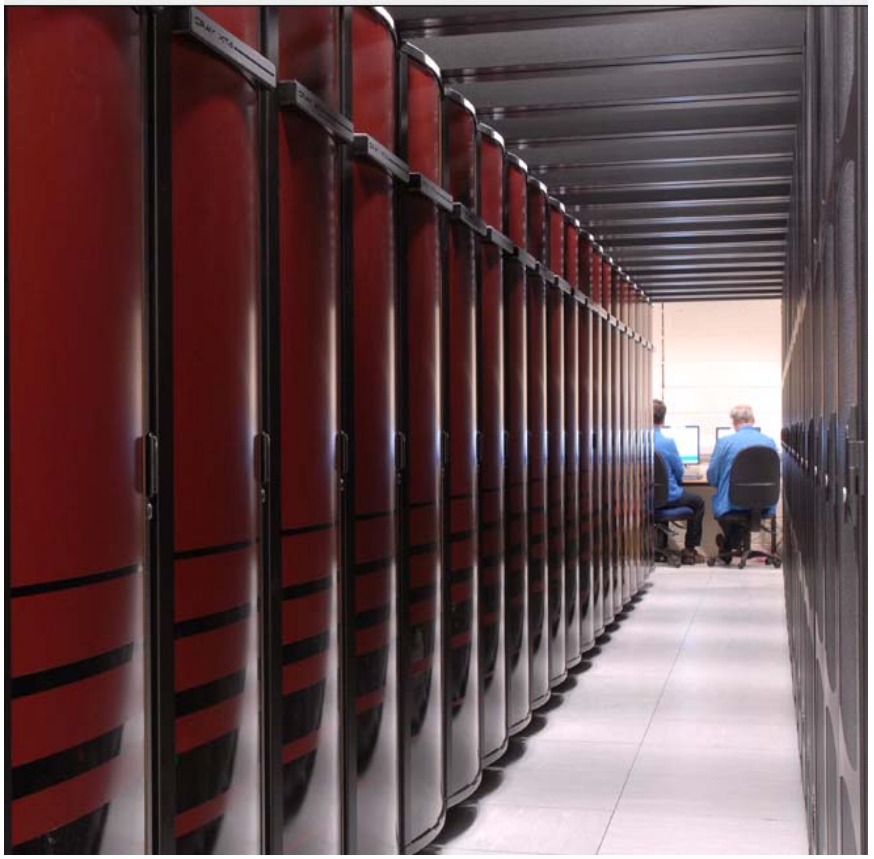
- Introduction
- Systems used
  - Cray XT4, IBM p575 (Power 5), IBM BlueGene/L
- All-to-All performance on HECToR and in comparison
- FFTs using multi dimensional virtual processor grids
  - Changing the grid extensions
  - Effect of placement on the multicore nodes
  - Task placement on the meshed communication Network
- Conclusions

Introduction

- Fast Fourier Transformations (FFT) important in many scientific applications
- Hard to parallelise on large numbers of tasks
- Distribute D dimensional FFT on processor grids of dimension up to D-1
- Requires all-to-all type communications

HECToR (Cray XT4)

- Newest national service in the UK
- Cray XT4 architecture
- 5664 dual core Opteron nodes
- 11328 cores, 2.8 GHz
- 6 GB memory/node
- 63.6 Tflop/s peak
- 54.6 Tflop/s linpack
- Mesh network: 20x12x24
  open in 12 direction
- Link speed 7.6 GB/s (Cray pub.)
- Bi-sectional BW: 3.6 TB/s



HPCx (IBM p575 Power5)

- National HPC service for the UK
- 160 IBM eServer p575, 16-way SMP nodes
- 2560 IBM Power 5 1.5 GHz processors
- IBM HPS Interconnect (aka. Federation)
- Bandwidth: 138 MB/s per IMB Ping-Ping pair, 2 full nodes
- 15.4 Tflop/s Peak, 12.9 Tflop/s Linpack
- 32 GB Memory/node

BlueSky (BlueGene/L)

- The University of Edinburgh
- IBM BlueGene/L
- 1024 IBM PowerPC 440 dual core nodes, 700 MHz
- 5.7 Tflop/s peak
- 4.7 Tflop/s Linpack
- Torus: 8x8x16, 8x8x8
  Mesh: 4x4x8, 2x4x4
- Link speed: 148 MB/s
- Bi-sectional BW: 18.5 GB/s

Bi-section Bandwidth

- Potential bottleneck for all-to-all communication:

  Bi-sectional bandwidth

  $t_{av} \geq D_T/(4B) = mn^2/(4B)$

- Effective bi-sectional bandwidth

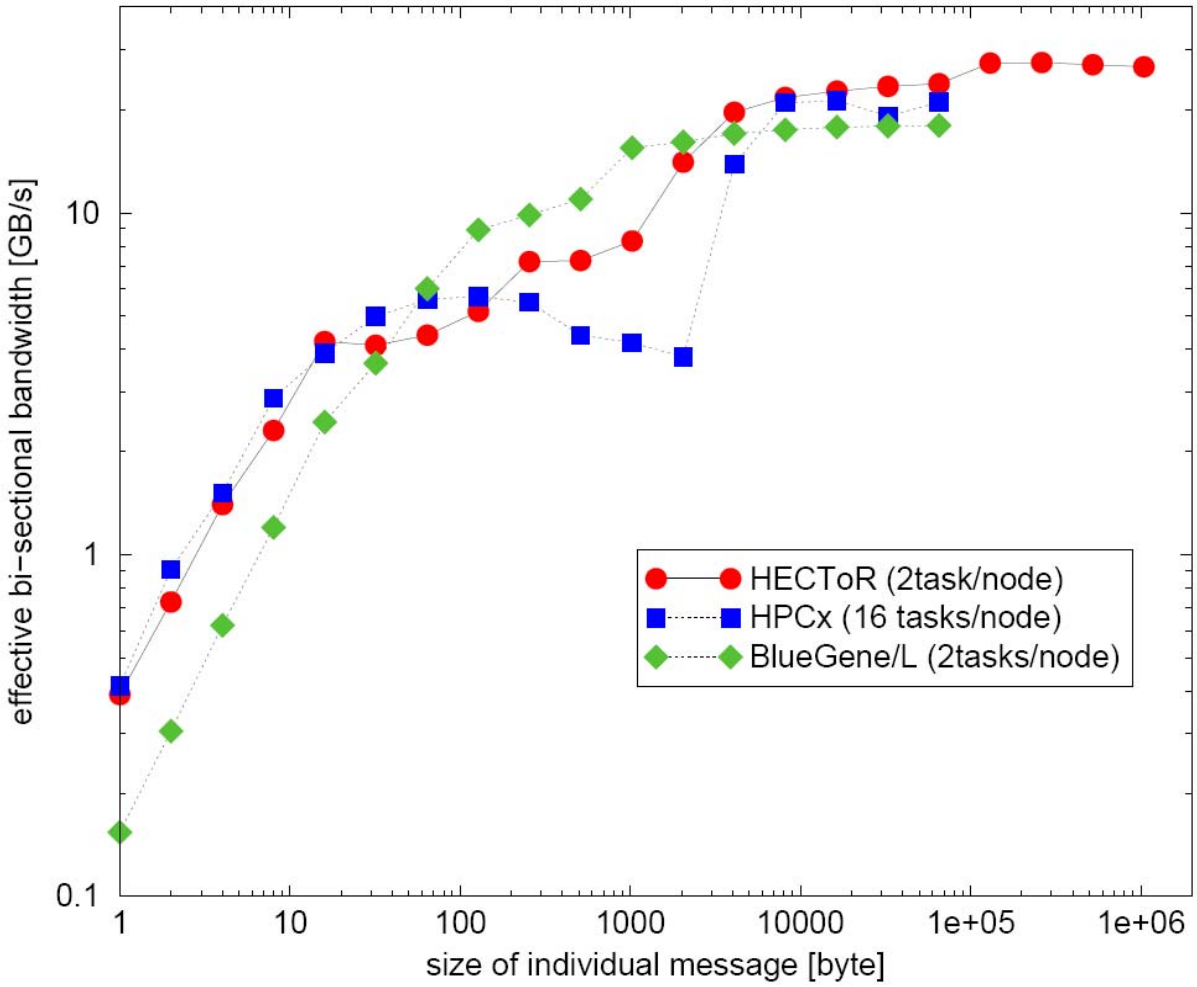
  $B_{eff} = D_T/(4t_{av}) = mn^2/(4t_{av})$

- Bi-sectional bandwidth (HW) on meshed (toroidal) network:

  Number of links cut, multiplied with link speed

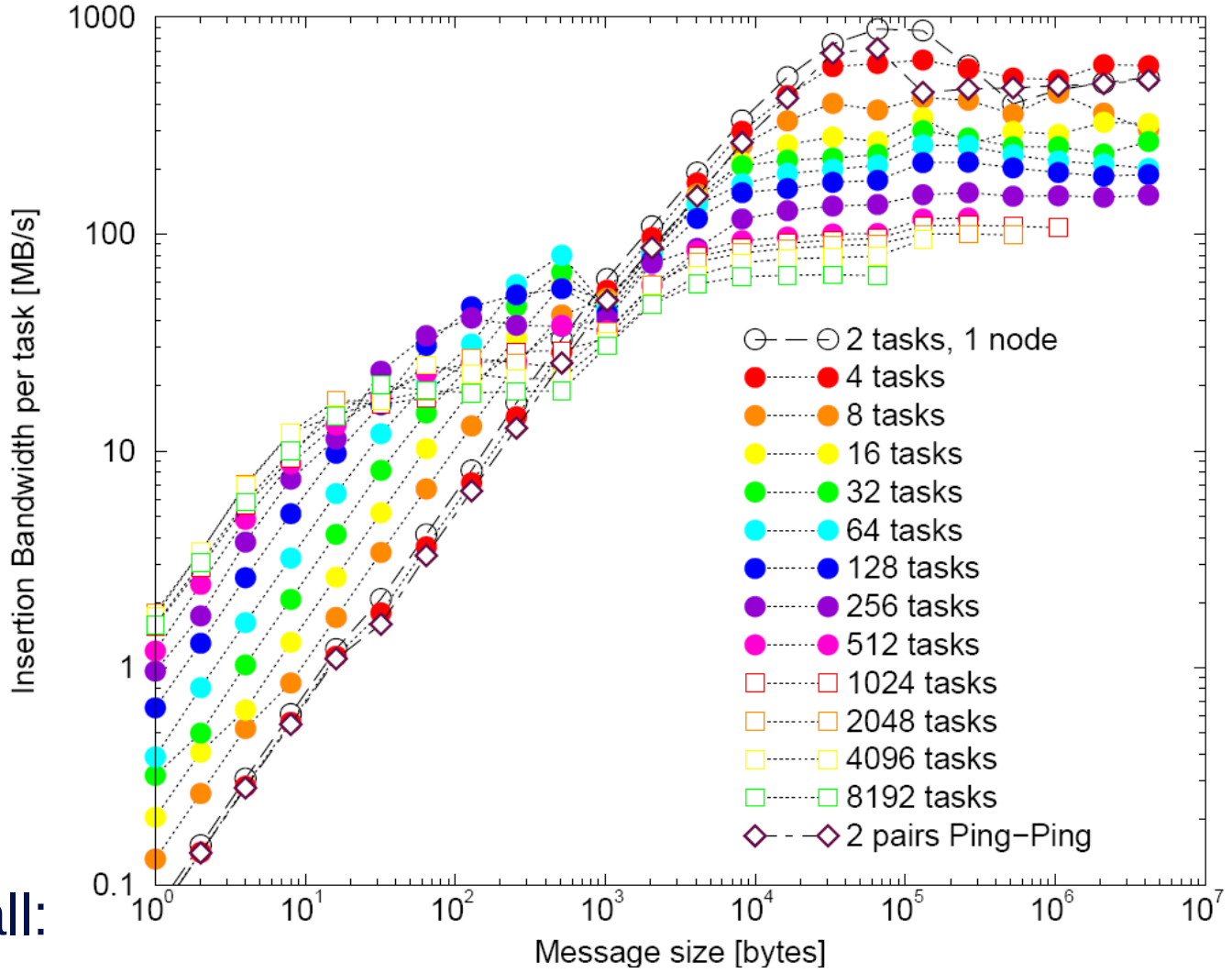
How does it compare?

- 1024 task All-to-all
- IMB v 3.0
- Compare best runs
- Complex double word: 16 byte
- Best results:
  - HECToR: 27.5 GB/s
  - HPCx: 21.3 GB/s
  - BlueGene/L: 18.1 GB/s



All-to-All performance on HECToR

- Intel MPI Bmark Version 3.0
- Insertion BW per task: $l_t = m(n-1)/t_{av}$
- Three Regions:
  - Below 1 kB
  - Up to 128 kB
  - Above 128 kB
- Low task all-to-all: similar performance to Ping-Ping



What is limiting the all-to-all?

- Comparing results for 4096 node All-to-all (73% of HECToR)

	Bi-section	**Insertion point**
Link speed: Datasheet value	7.6 GB/s	6.4 GB/s
Link speed: Ping-Ping 2 tasks/node	1.4 GB/s	1.4 GB/s
Link speed: Ping-Ping 1 task/node	1.4 GB/s	1.4 GB/s
Number of links	20 × 24 = 480	4096
Theoretical from Cray datasheet	**3.6 TB/s**	25.6 TB/s
Scaled bandwidth from Ping-Ping, 2 t/n	**0.66 TB/s**	5.6 TB/s
Scaled bandwidth from Ping-Ping, 1 t/n	**0.66 TB/s**	5.6 TB/s
Bandwidth from all-to-all, 2 t/n	**0.13 TB/s**	0.51 TB/s
Bandwidth from all-to-all, 1 t/n	**0.21 TB/s**	0.85 TB/s

- **Answer:** Not clear, but result is short of expectation!

FFT of a three dimensional array

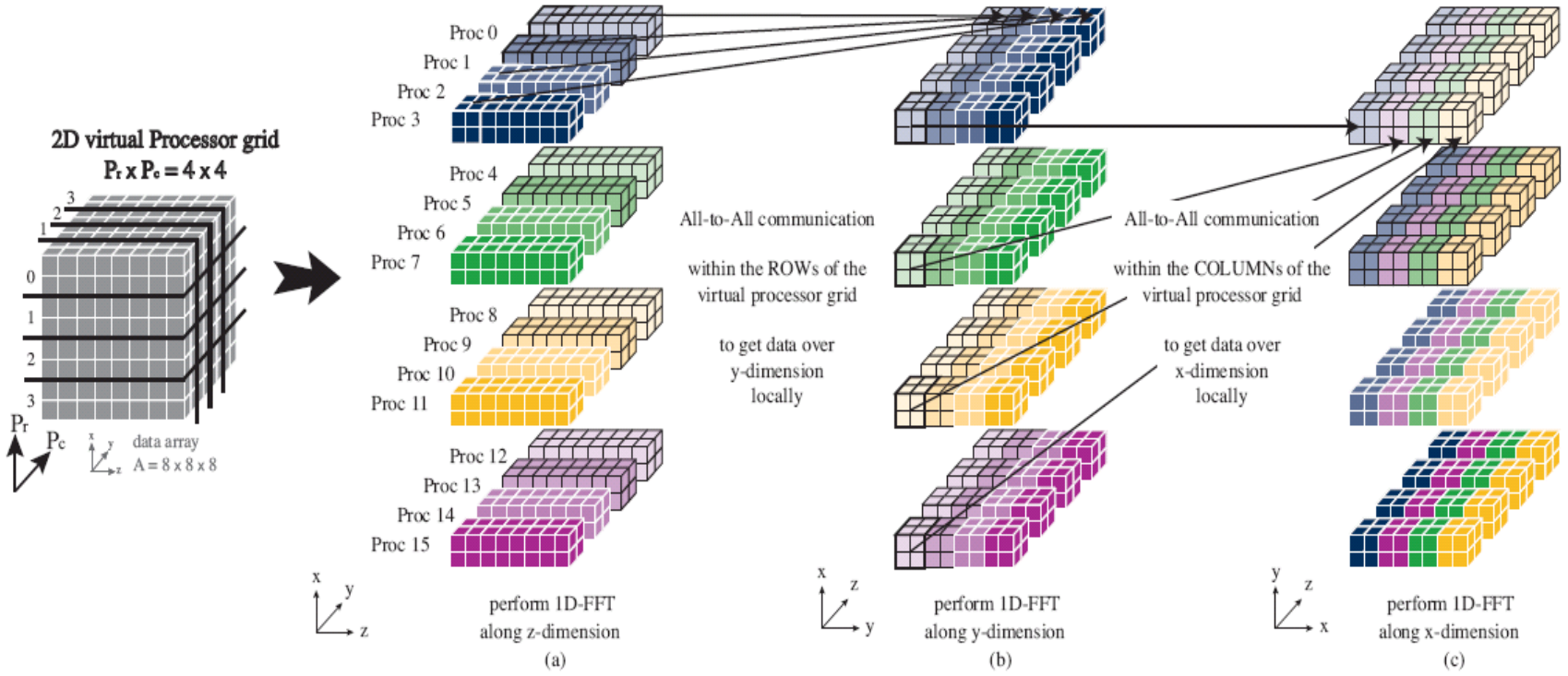
- Fourier Transformation of array X(x,y,z)

$$\tilde{X}(k_x, k_y, k_z) := \sum_{x=0}^{N_x-1} \left(\sum_{y=0}^{N_y-1} \left(\sum_{z=0}^{N_z-1} X(x,y,z) \exp\left(\frac{2\pi \mathrm{i}}{N_z} k_z z\right) \right) \exp\left(\frac{2\pi \mathrm{i}}{N_y} k_y y\right) \right) \exp\left(\frac{2\pi \mathrm{i}}{N_x} k_x x\right)$$

- Parallelise using 2-D virtual processor grid
  1. Perform FFT in z-direction
  2. Groups of All-to-all in the rows: y-direction task local
  3. Perform FFT in y-direction
  4. Groups of All-to-all in the columns: x-direction task local
  5. Perform FFT in x-direction

Illustration of the Algorithm

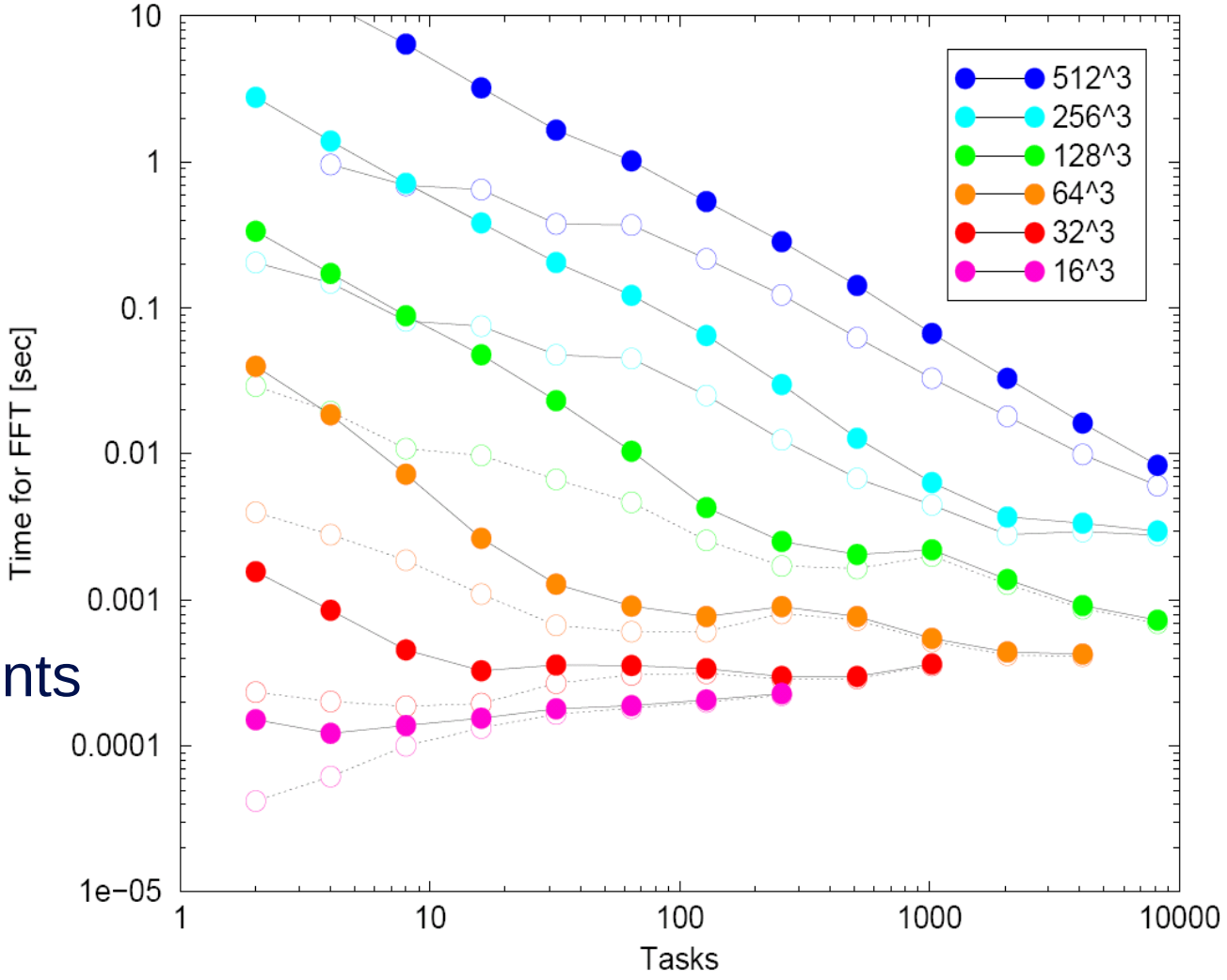
- Example: 8x8x8 problem on 16 task



- **Remark:** inserted data almost independent of virtual proc. grid, apart from own data effects

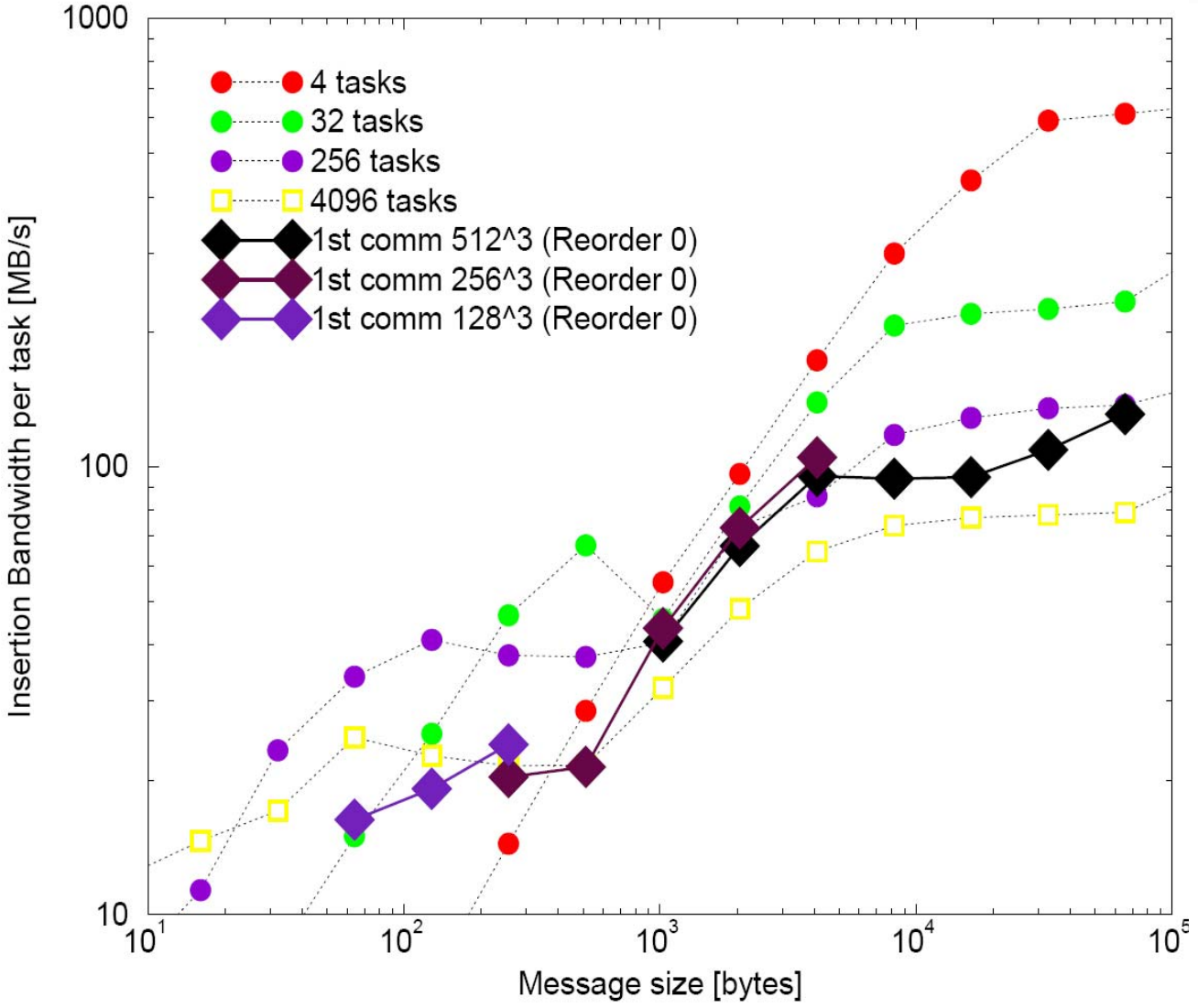
Parallel FFT performance on HECToR

- Closed symbols:
  Total time
- Open symbols:
  Comm. Time
- Poor
  "intermediate" points
  1 kB messages



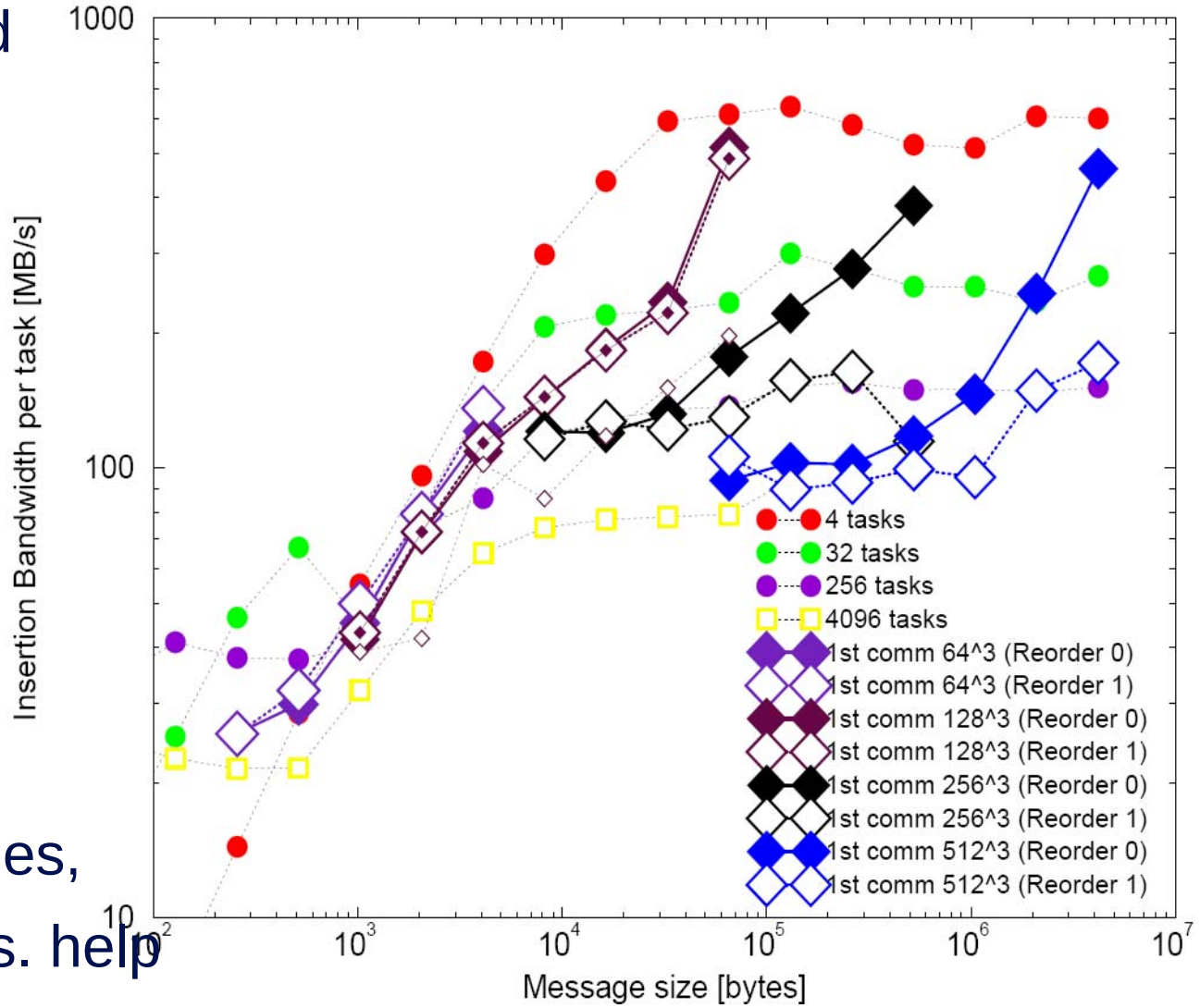
Effect of decomposition on 4096 tasks

- Change Proc. grid
  8x512 to 512x8
- 1st comm phase
- Intra-node little effect
- Performance similar to large task all-to-all
- Indication of congestion?



Effect of decomposition on 256 tasks

- Change proc. grid 2x128 – 128x2
- 1st comm. phase
- Results in range of global All-to-all
- For large messages, inter-node comms. help

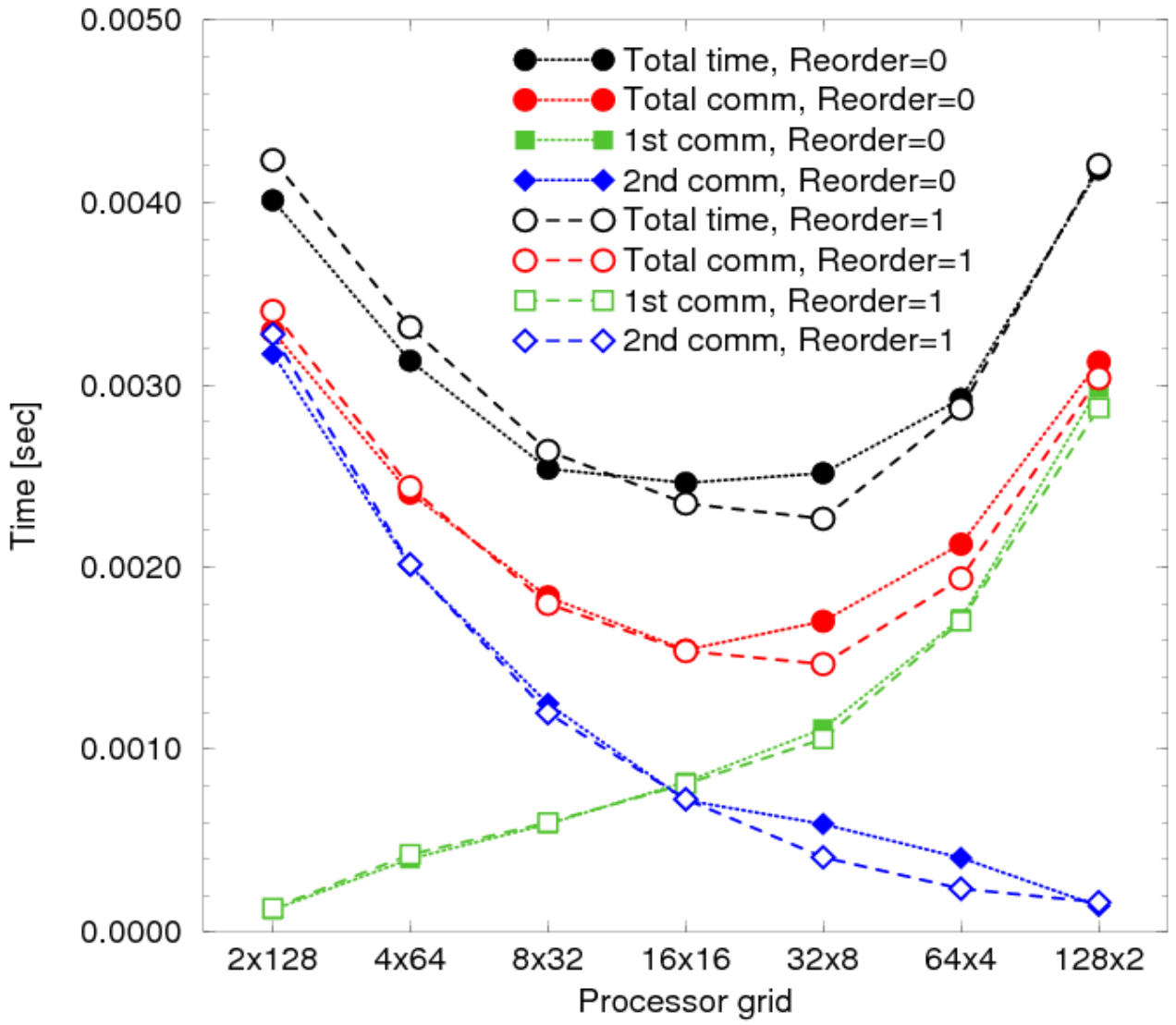


Communication time

- Applications care about time
- Small communicators:
  - Relation between the two metrics distorted due to “own data”
- Discuss two characteristic cases with respect to time

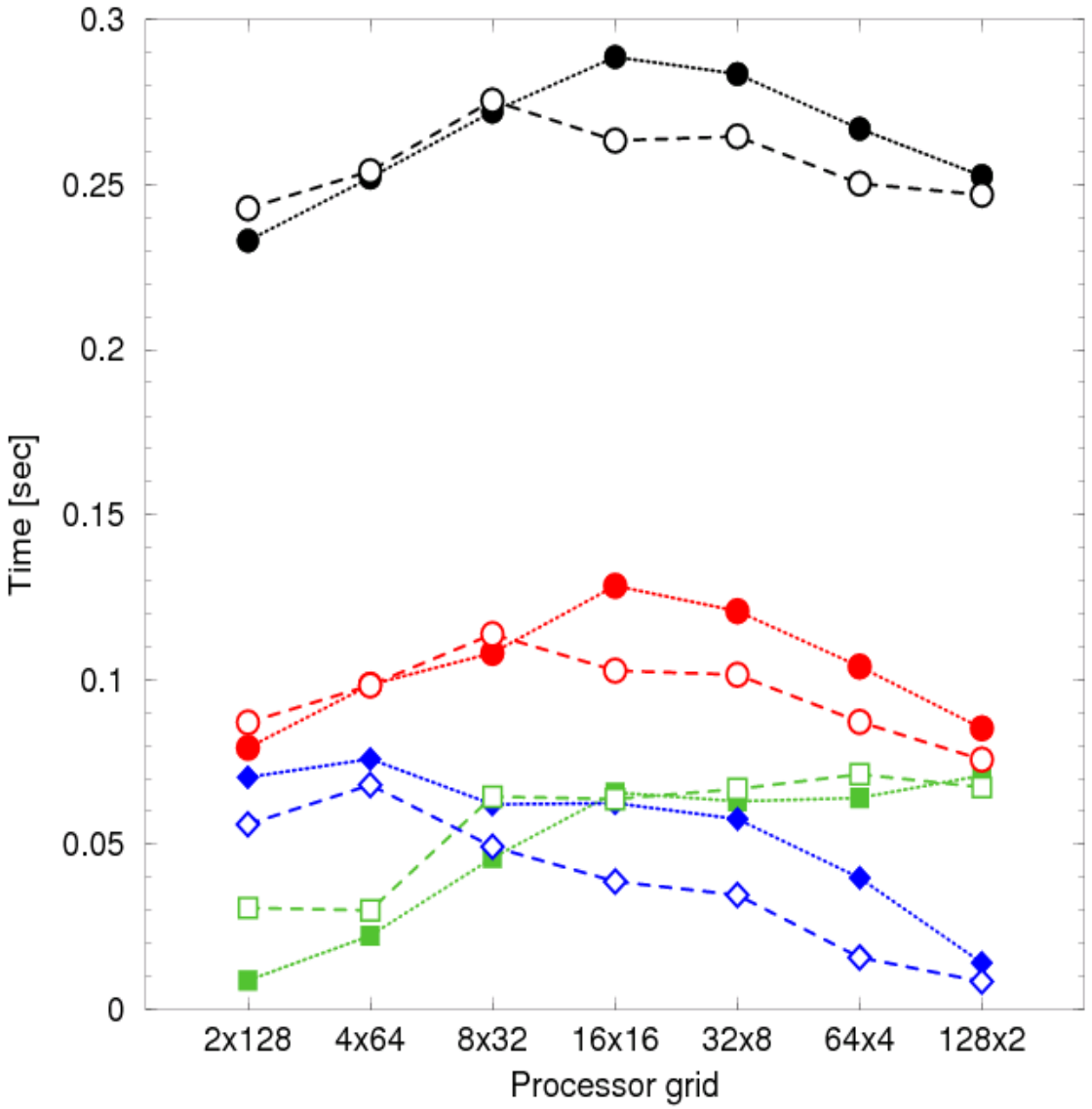
Timings for 128^3 on 256 tasks

- Penalty for large communicators
  - Bandwidth
  - Data amount
- The other communication can't make up
- 16x16 best
- Little effect of intra node communication



Timings for 512^3 on 256 tasks

- Bandwidth almost in-dependent of message size
- Small communicators insert less data
- Intra node comms beneficial
- For total time effect almost cancels
- Best to use 2x128 or 128x2



Task placement on a meshed Network

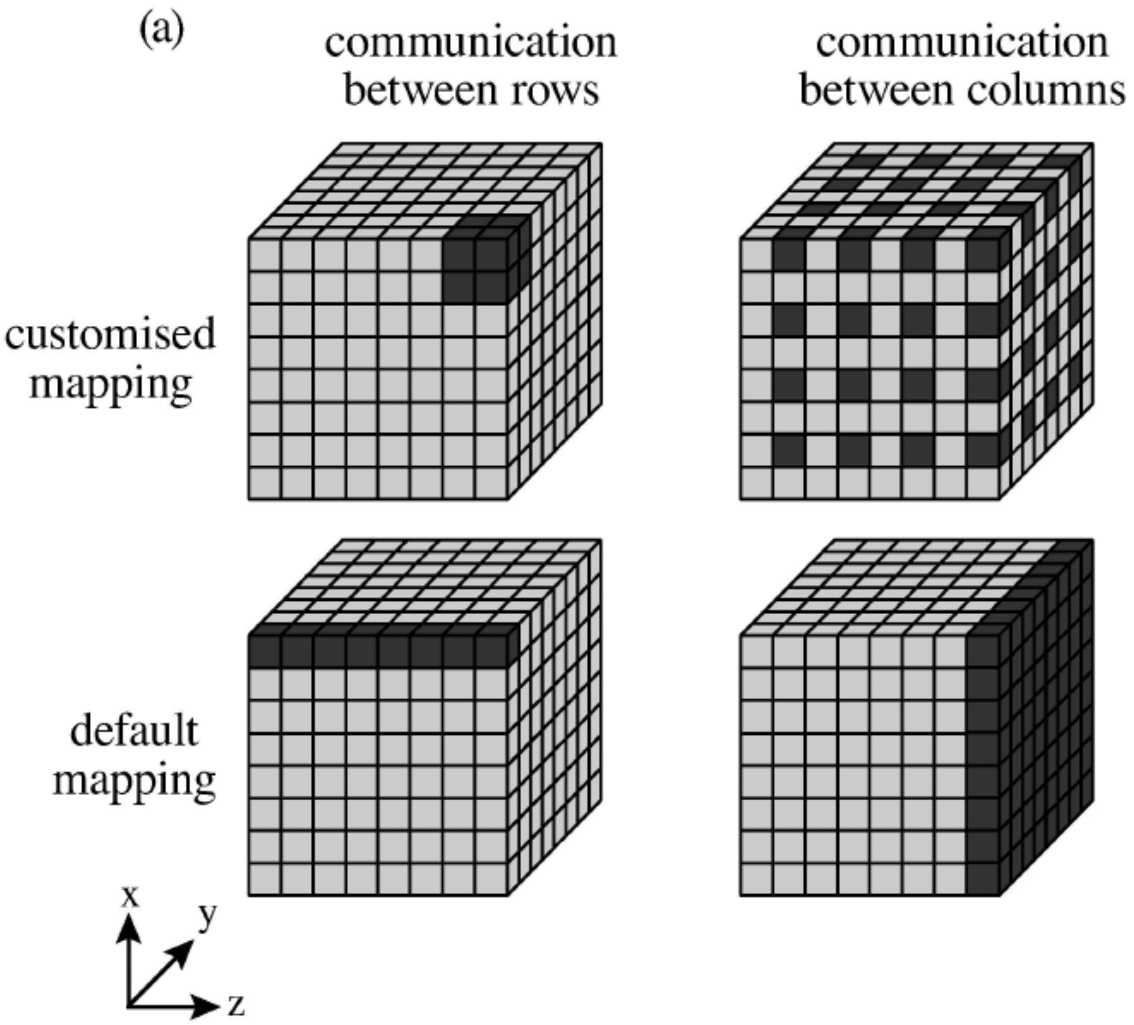
- Cray XT architecture: limited user control on task placement
  - Placement with respect to multi-core chips
  - No control on placement on the meshed network
  - Schedules individual nodes

- Use a Bluegene/L for a case study
  - Schedules jobs on dense cuboidal partitions (no holes!)
  - Offers full control of task placement (re. multi core **and** mesh position)
  - Downside: Scheduling constraints

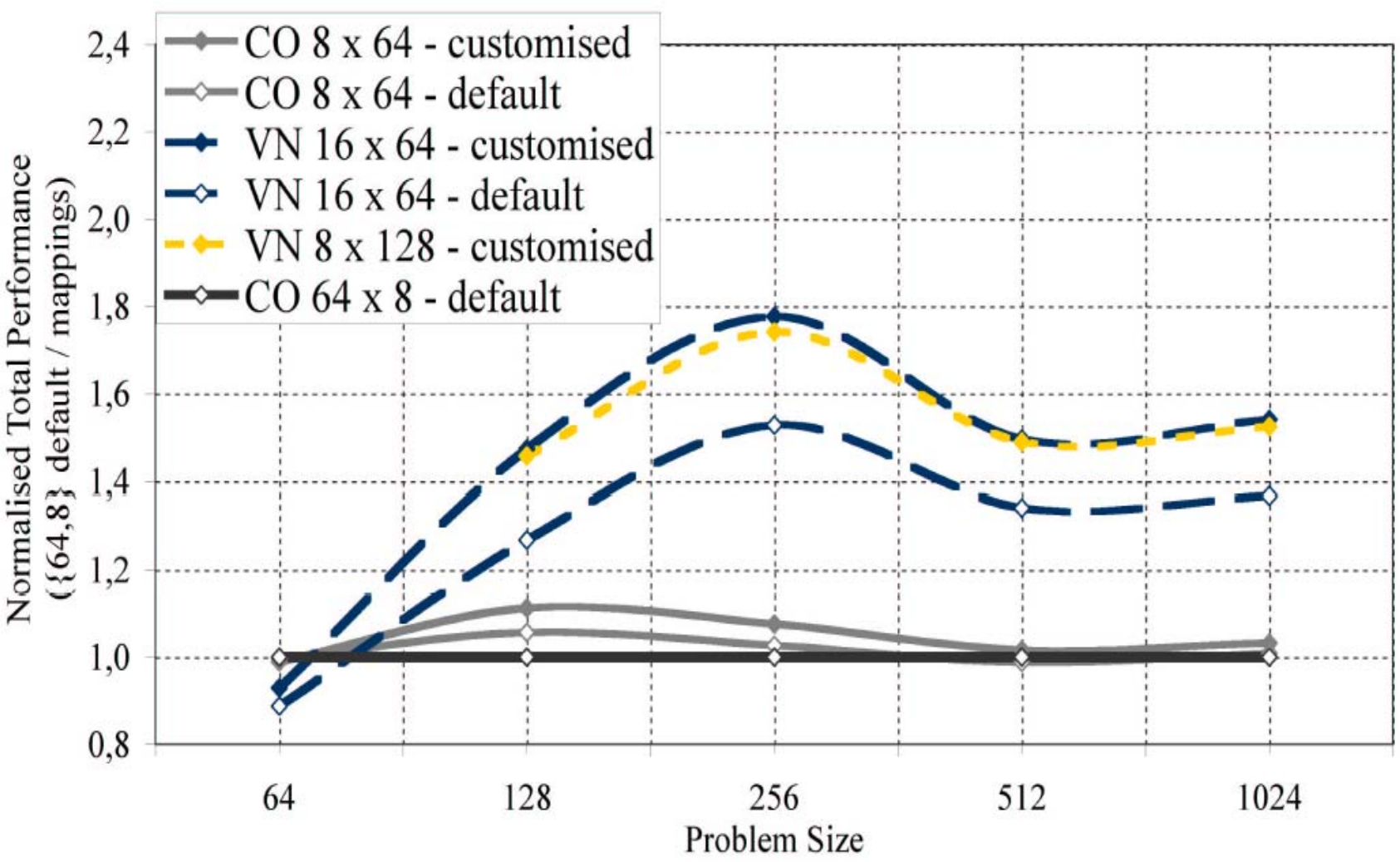
- Derived a model from bi-sectional BW considerations
  - Place rows of the processor grid on small cubes should work best

Illustration of the maps

- Processor grids:
  - 8x64 in CO mode
  - 16x64 in VN mode
  - 8x128 in VN mode
- Map rows on cubes
- Columns map to extended objects
- Default: sticks & planes
- All maps but 2^3-cube offer same bi-sectional Bandwidth
- Idea for cube: Many mini-BG/L



Normalised performance



- Little benefit in CO mode, small cube does't perform ☹
- Works well in VN mode, boost of up to 16% ☺

Conclusion

- Cray XT4 faster than IBM Power5 HPS and BlueGene/L for 1024 tasks, but only just and not for every message size
- Global all-to-all on the Cray XT4 for thousands of tasks does not live up to expectations from marketing materials and Ping-Ping results
- Performance of all-to-all in subgroups, similar to global all-to-all
- For large task count performance similar to a single all-to-all of the total size and not the size of the subgroup
  - Indicating a congestion problem?

Conclusion (cont.)

- Little overall effect from intra node communication
- Placing rows onto cubes inside the mesh gives performance advantage (BlueGene/L)
- On the Cray XT4 such placement is not supported by the system software
- If it was, it might help to overcome the performance problems for messages > 1 kB on large task counts (many mini XTs)

Acknowledgement

- Mark Bull (EPCC) and Stephen Booth (EPCC)

- David Tanqueray, Jason Beech-Brandt, Kevin Roy and Martyn Foster (Cray)