science + computing

A Bull Group Company

Runtime Performance Optimizations for an OpenFOAM Simulation

Dr. Oliver Schröder
HPC Services and Software

science + computing ag
IT Service and Software Solutions for Complex Computing Environments
Tuebingen | Muenchen | Berlin | Duesseldorf

Many Thanks to…

- Applications and Performance Team
  - Madeleine Richards
  - Rafael Eduardo Escovar Mendez

- HPC Services and Software Team
  - Fisnik Kraja
  - Josef Hellauer

Dr. Oliver Schröder | o.schroeder@science-computing.de

Overview

science + computing
A Bull Group Company

- Introduction
- The Test Case
- The Benchmarking Environment
- Initial Results (the starting point)

- Dependency Analysis
  - CPU Frequency
  - Interconnect
  - Memory Hierarchy

- Performance Improvements
  - MPI Communications
    - ✓ Tuning MPI Routines
    - ✓ Intel MPI vs. Bull MPI
  - Domain Decomposition

- Conclusions

Dr. Oliver Schröder | o.schroeder@science-computing.de

Introduction

science + computing
A Bull Group Company

- How can we improve application performance?
  - By using the resources efficiently
  - By selecting the appropriate resources
- In order to do this we have to:
  - Analyze the behavior of the application
  - And then tune the runtime environment accordingly
- In this study we:
  - Analyze the dependencies of an OpenFOAM Simulation
  - Improve the performance of OpenFOAM by
    - Tuning MPI communications
    - Selecting the appropriate domain decomposition

Dr. Oliver Schröder | o.schroeder@science-computing.de

The Test Environment

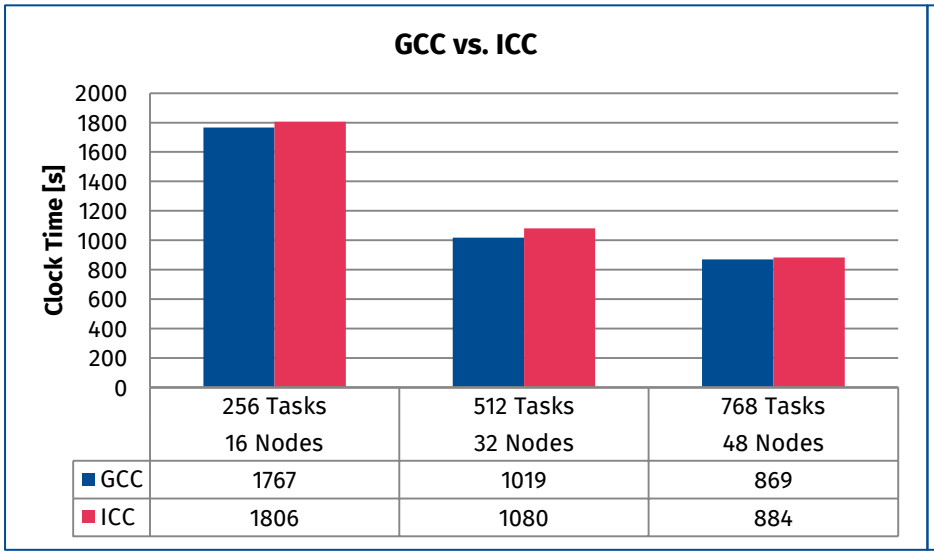
Architect of an Open World™

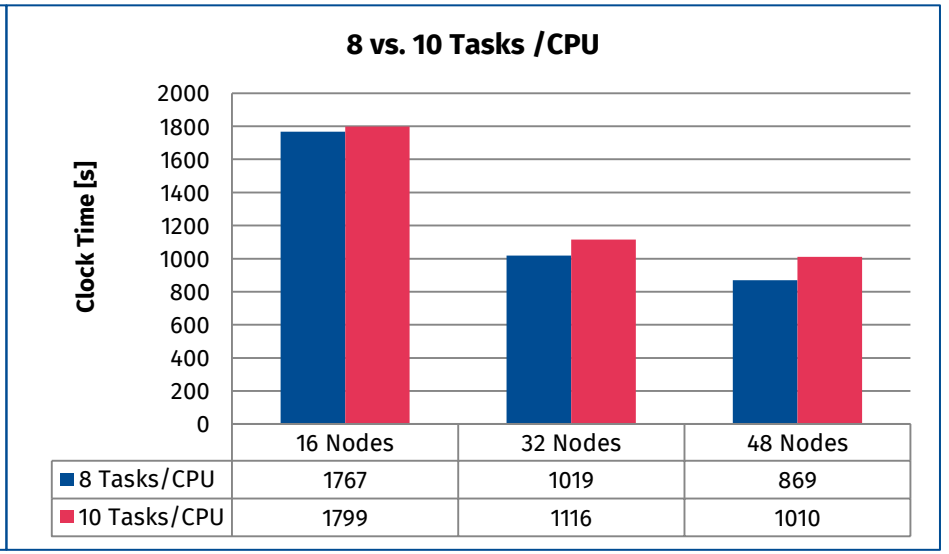
science + computing | A Bull Group Company

Compute Cluster	Sid	Robin
Hardware		
Processor	Intel E5-2680 V2 Ivy-Bridge	Intel E5-2697 V2 Ivy-Bridge
Frequency	2.80 GHz	2.70 GHz
Cores per processor	10	12
Sockets per node	2	2
Cores per nodes	20	24
Cache	25 MB (L3), 10x256 KB (L2)	30 MB (L3), 12 x 256 KB (L2)
Memory	64 GB	64 GB
Frequency of memory	1866 MHz	1866 MHz
IO – FS	NFS over IB	NFS over IB
Interconnect	IB FDR	IB FDR
Software		
OpenFOAM	2.2.2	2.2.2
Intel C Compiler	13.1.3.192	13.1.3.192
GNU C Compiler	4.7.3	4.7.3
Compiler Options	-O3 –fPIC –m64	-O3 –fPIC –m64
Intel MPI	4.1.0.030	4.1.0.030
Bull MPI	1.2.4.1	1.2.4.1
SLURM	2.6.0	2.5.0
OFED	1.5.4.1	1.5.4.1

Dr. Oliver Schröder | o.schroeder@science-computing.de

Initial Results on SID (E5-2680v2) (1)





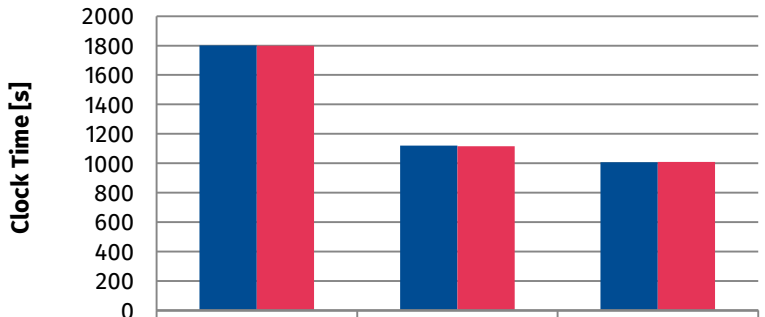
- GCC(IMPI) vs. ICC (IMPI)
  - Better performance is obtained with the OpenFOAM version compiled with GCC
  - During the development and optimization of OpenFOAM mainly GCC is used
- 8 vs. 10 Tasks / CPU
  - Better Performance with 8 Tasks/CPU. The reasons might be:
    - With fewer tasks we use fewer cores in each CPU, having so *more cache* and *more memory bandwidth per core*.
    - The *domain decomposition* is different. We will see later the impact of Domain Decomposition on Performance

Dr. Oliver Schröder | o.schroeder@science-computing.de

Initial Results on SID (E5-2680v2) (2)

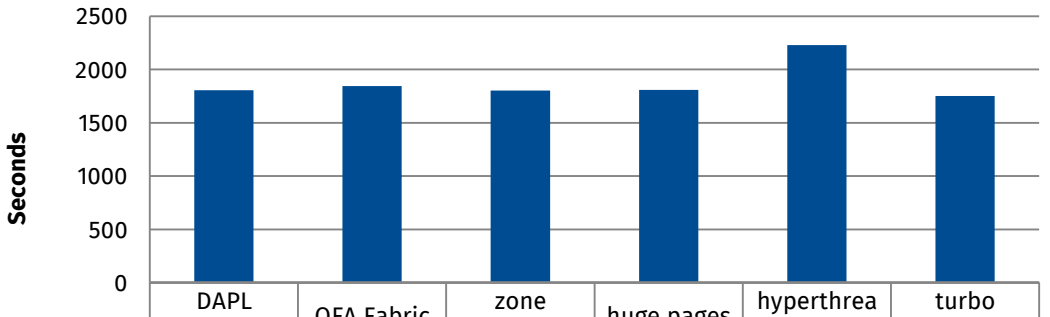
science + computing
A Bull Group Company

OpenFOAM: CPU Binding



	320 Tasks 16 Nodes	640 Tasks 32 Nodes	960 Tasks 48 Nodes
No_CPU_Binding	1801	1121	1007
With_CPU_BINDING	1799	1116	1010

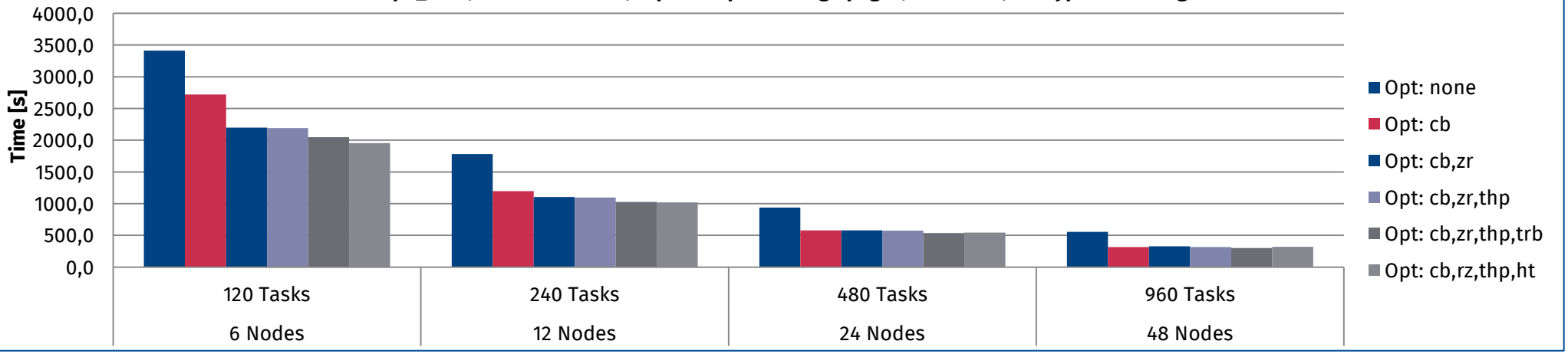
OpenFOAM Tests on 16 Nodes



	DAPL Fabric 16 Nodes 320 Tasks	OFA Fabric 16 Nodes 320 Tasks	zone reclaim 16 Nodes 320 Tasks	huge pages 16 Nodes 320 Tasks	hyperthreading 16 Nodes 640 Tasks	turbo boost 16 Nodes 320 Tasks
Clock Time [s]	1804	1845	1802	1807	2229	1750

Example results with Star-CCM+

(cb=cpu_bind, zr=zone reclaim, thp=transparent huge pages, trb=turbo, ht=hyperthreading)



Dr. Oliver Schröder | o.schroeder@science-computing.de

Initial Results on SID (E5-2680v2) (3)

science + computing
A Bull Group Company

- CPU Binding
  - With other applications we often observed performance improvements when we bind the Tasks to specific CPUs (especially when Hyperthreading=ON)
  - This seems not to be the case for this OpenFOAM simulation
- We applied also other optimization techniques but we observed improvement only in the case of Turbo Boost.
  - Actually with Turbo Boost we expected more than 3% improvement.
- We really need to analyze the dependencies of OpenFOAM in order to understand this behavior.

Dr. Oliver Schröder | o.schroeder@science-computing.de

Overview

science + computing
| A Bull Group Company

- Introduction
- The Test Case
- The Benchmarking Environment
- Initial Results (the starting point)

- Dependency Analysis
  - CPU Frequency
  - Interconnect
  - Memory Hierarchy

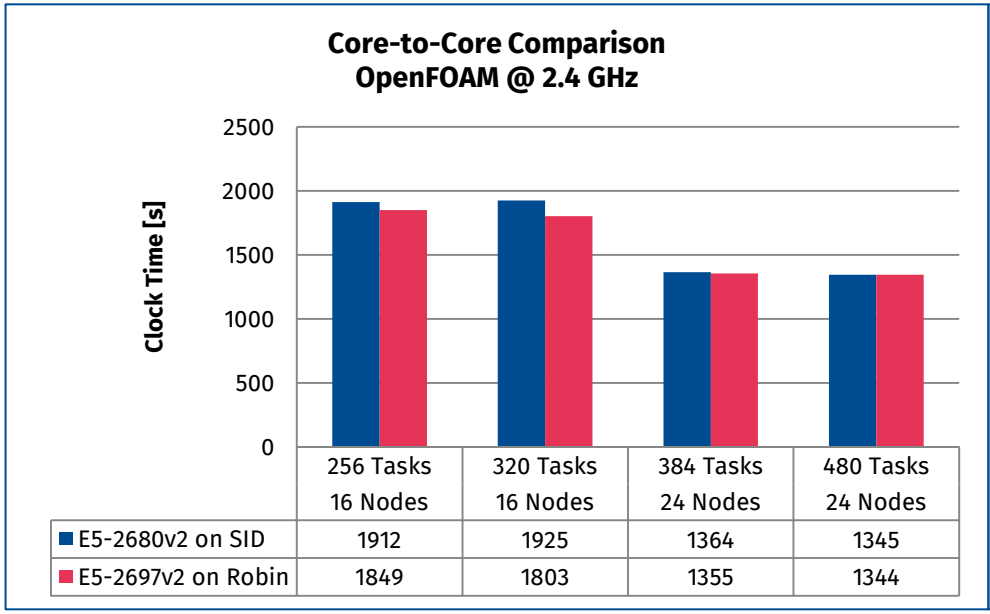
- Performance Improvements
  - MPI Communications
    - ✓ Tuning MPI Routines
    - ✓ Intel MPI vs. Bull MPI
  - Domain Decomposition

- Conclusions

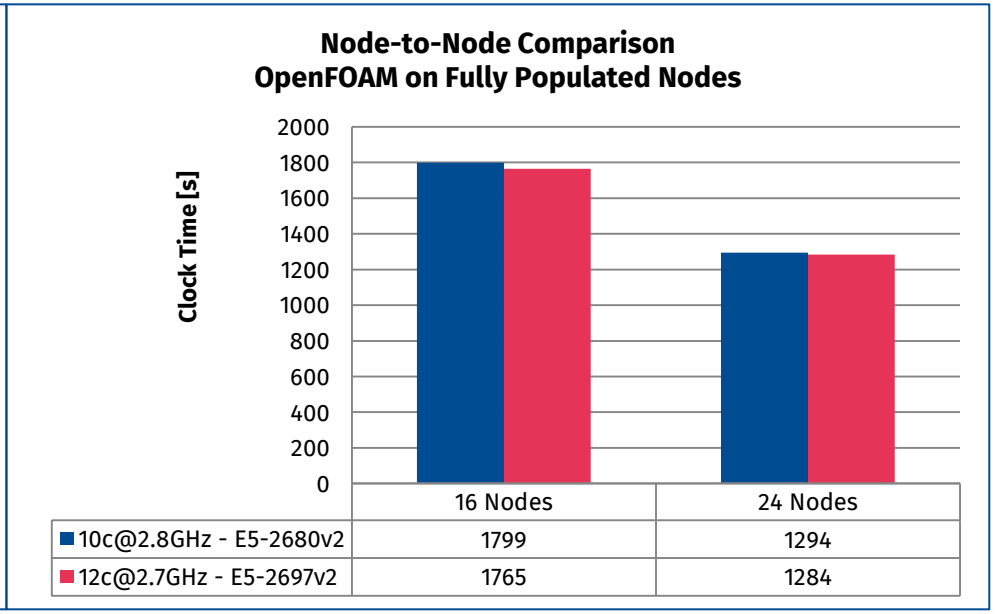
Dr. Oliver Schröder | o.schroeder@science-computing.de

CPU Comparison
E5-2680v2 vs. E5-2697v2

science + computing
A Bull Group Company



	256 Tasks 16 Nodes	320 Tasks 16 Nodes	384 Tasks 24 Nodes	480 Tasks 24 Nodes
E5-2680v2 on SID	1912	1925	1364	1345
E5-2697v2 on Robin	1849	1803	1355	1344



	16 Nodes	24 Nodes
10c@2.8GHz - E5-2680v2	1799	1294
12c@2.7GHz - E5-2697v2	1765	1284

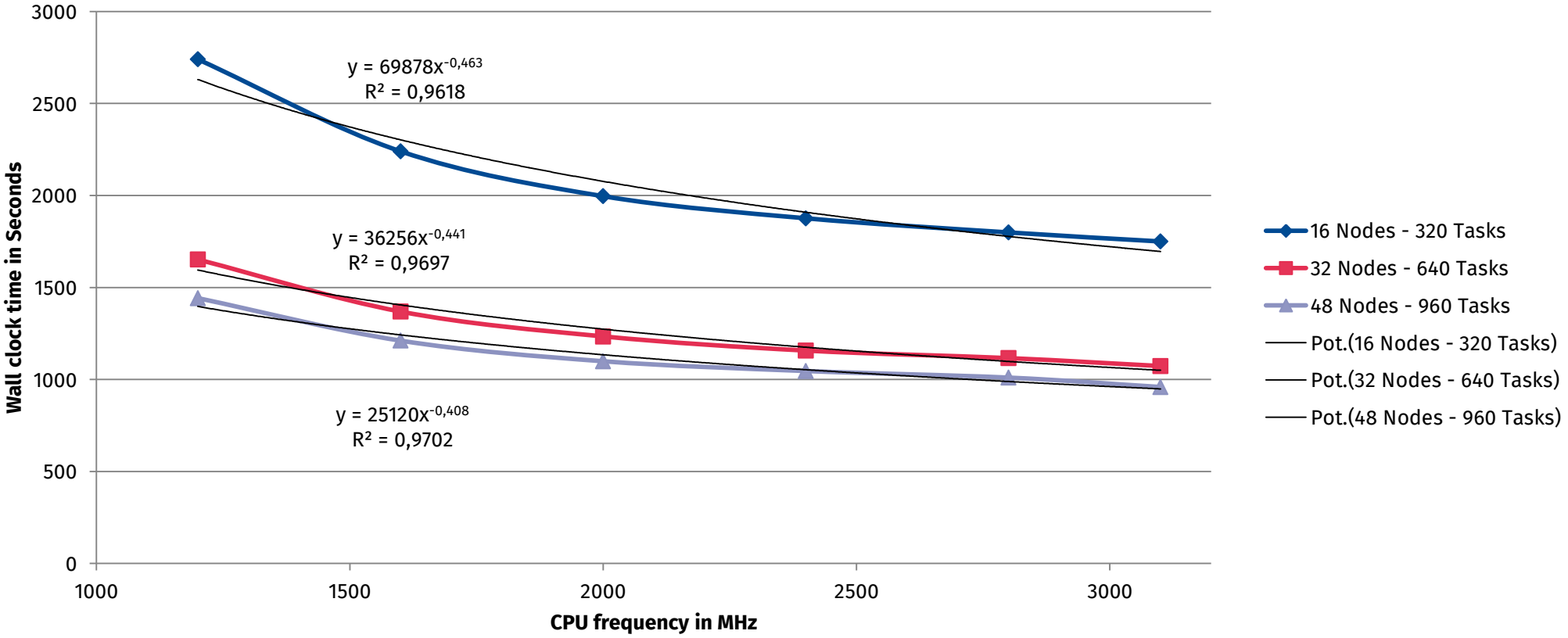
- Core-to-Core Comparison
  - 16 nodes: better on E5-2697v2 CPUs considering the *additional 5 MB* of L3 cache
  - 24 nodes: very small differences – MPI overhead hides the benefits from the cache
- Node-to-Node Comparison
  - 16 nodes: 2% better on E5-2697v2 CPUs
  - 24 nodes: <1% better on E5-2697v2 CPUs

Dr. Oliver Schröder | o.schroeder@science-computing.de

OpenFOAM Dependency on CPU Frequency

Tests on SID B71010c (E5-2680v2)

science + computing
A Bull Group Company

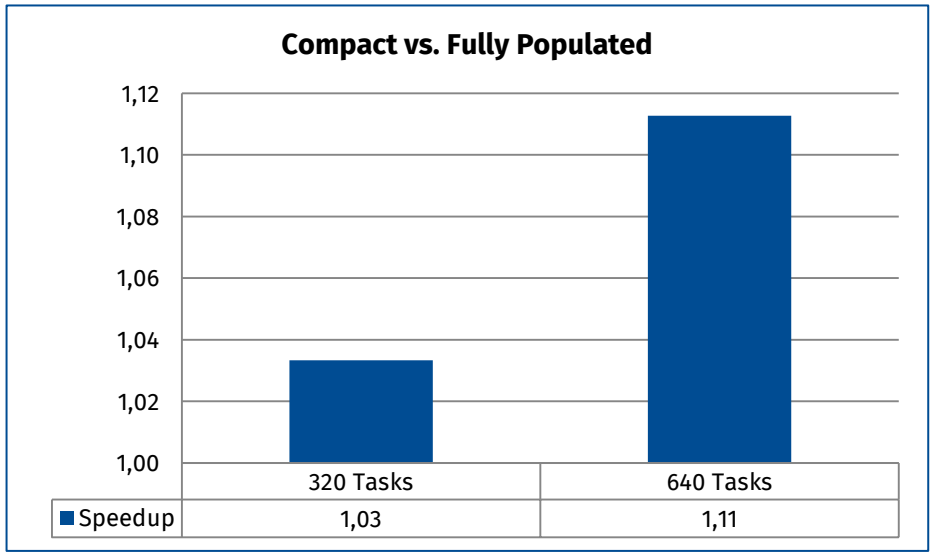


- Dependency on the CPU frequency only modest, approx 45%
- Beyond 2.5 GHz there is not much improvement

Dr. Oliver Schröder | o.schroeder@science-computing.de

OpenFOAM Dependency on Interconnect
Compact Task Placement vs. Fully Populated

science + computing
A Bull Group Company



- Test
  - *Fully Populated*: 20 Tasks/Node
    - 320 Tasks on 16 Nodes
    - 640 Tasks on 32 Nodes
  - *Compact*: 10 Tasks/Node (2x #Nodes)
    - 320 Tasks on 32 Nodes
    - 640 Tasks on 64 Nodes

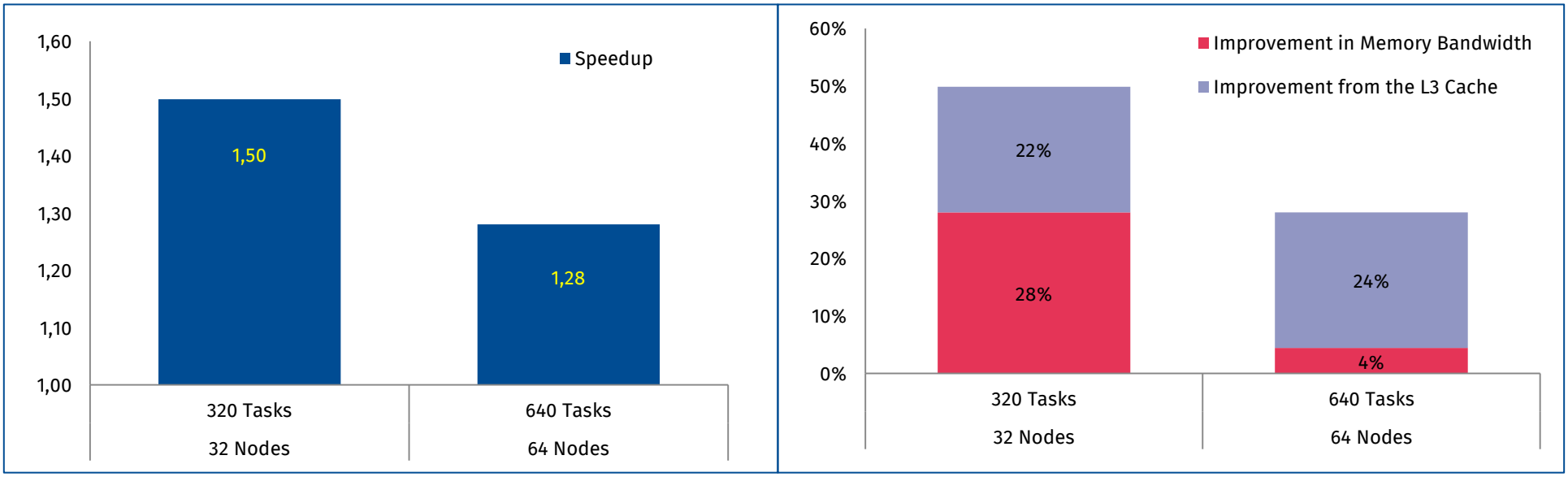
- With the *compact* task placement we bind all 10 tasks on *one of the CPUs* in each node to make sure that cache size and memory bandwidth stay the same. In this way we give to each task *more interconnect bandwidth*.
  - For tests with 320 tasks we obtain 3% improvement
  - For tests with 640 tasks we obtain 11% improvement

- Conclusion:
  - Dependency on Interconnect Bandwidth increases with the number of tasks (nodes)

Dr. Oliver Schröder | o.schroeder@science-computing.de

OpenFOAM Dependency on Memory Hierarchy
Scatter vs. Compact Task Placement

science + computing
A Bull Group Company



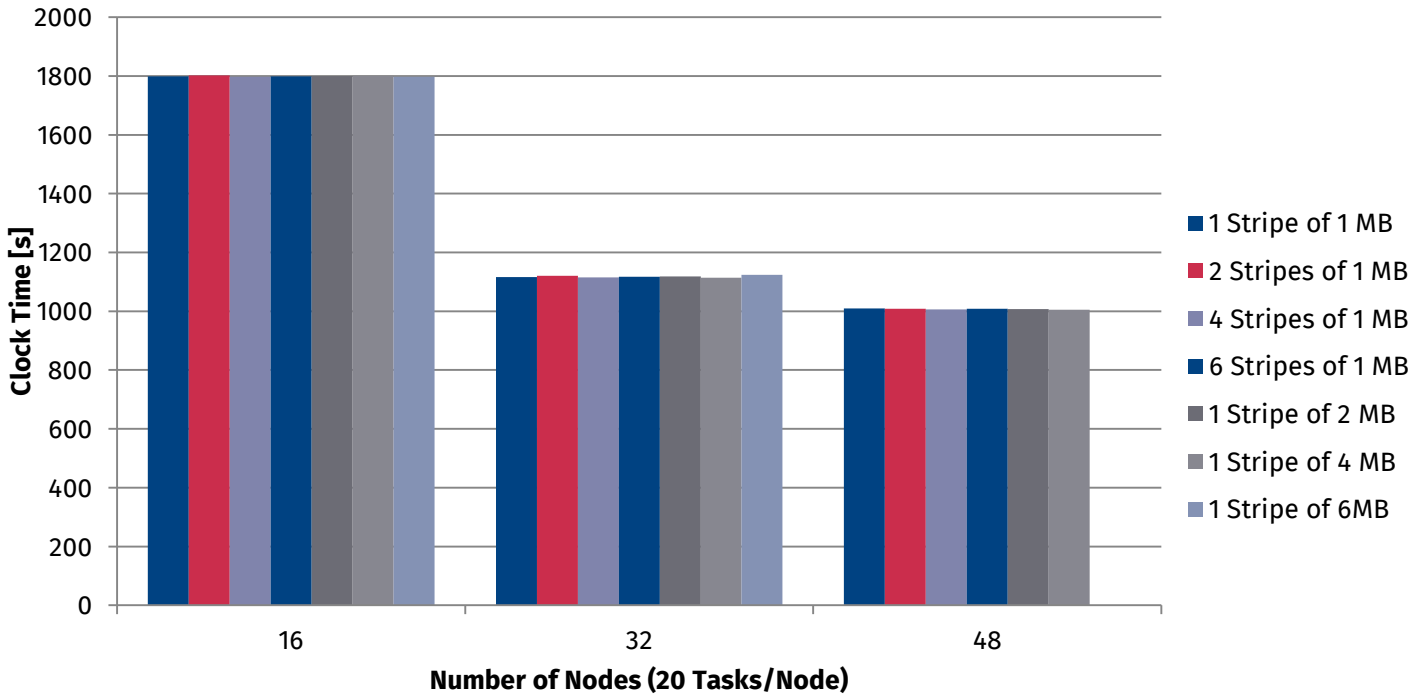
- The dependency on the memory hierarchy is *bigger* when running OpenFOAM *on a small number of nodes*. On more nodes, the *bottleneck is not anymore the memory hierarchy but the interconnect.*

- The impact of the *L3 cache* stays almost the *same* when doubling the number of nodes. However, the impact coming from the *memory throughput is lower*.

Dr. Oliver Schröder | o.schroeder@science-computing.de

OpenFOAM Dependency on Memory Hierarchy
Scatter vs. Compact Task Placement

science + computing
A Bull Group Company



- We tested OpenFOAM on Lustre with various
  - Stripe Counts
  - Stripe Sizes

- We could **not** obtain **any improvement**

Dr. Oliver Schröder | o.schroeder@science-computing.de

Overview

science + computing
| A Bull Group Company

- Introduction
- The Test Case
- The Benchmarking Environment
- Initial Results (the starting point)

- Dependency Analysis
  - CPU Frequency
  - Interconnect
  - Memory Hierarchy

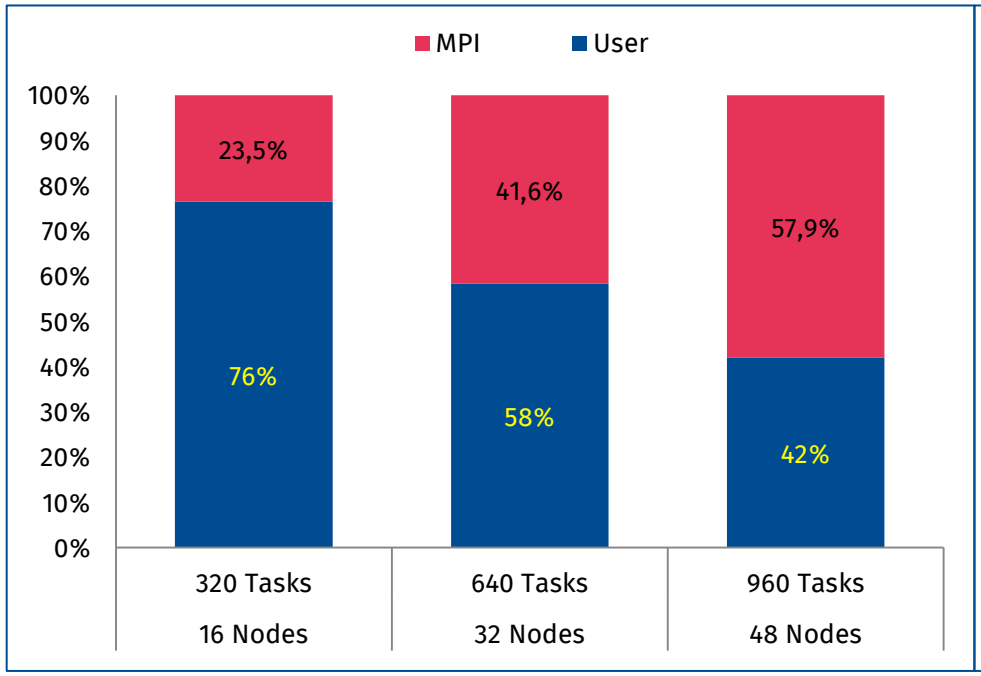
- Performance Improvements
  - MPI Communications
    - ✓ Tuning MPI Routines
    - ✓ Intel MPI vs. Bull MPI
  - Domain Decomposition

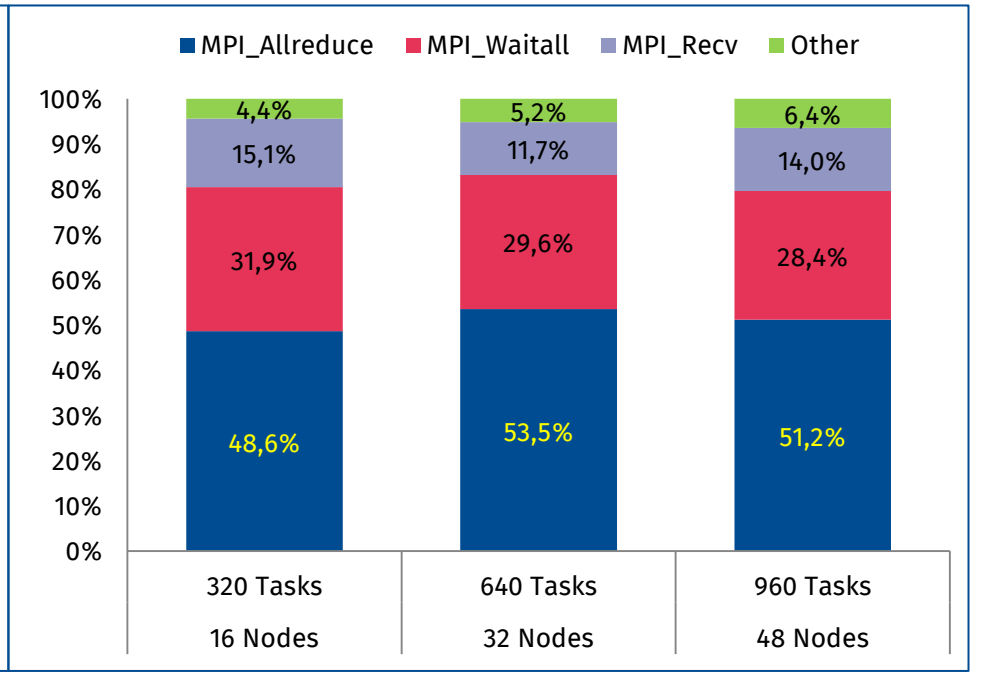
- Conclusions

Dr. Oliver Schröder | o.schroeder@science-computing.de

MPI Profiling (with Intel MPI)
Tests on SID B71010c (E5-2680v2)

science + computing
A Bull Group Company





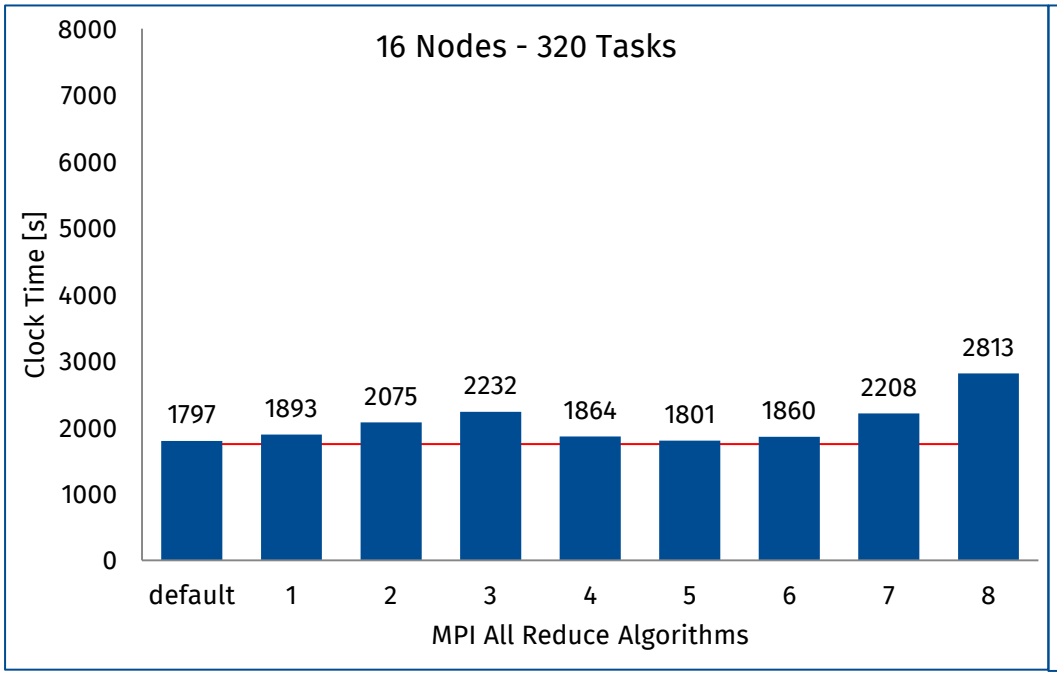
- The portion of time spent in MPI increases with the number of nodes
- Looking at the MPI Calls
  - ~ 50 % of MPI Time is spent on MPI_Allreduce
  - ~ 30 % of MPI Time is spent on MPI_Waitall

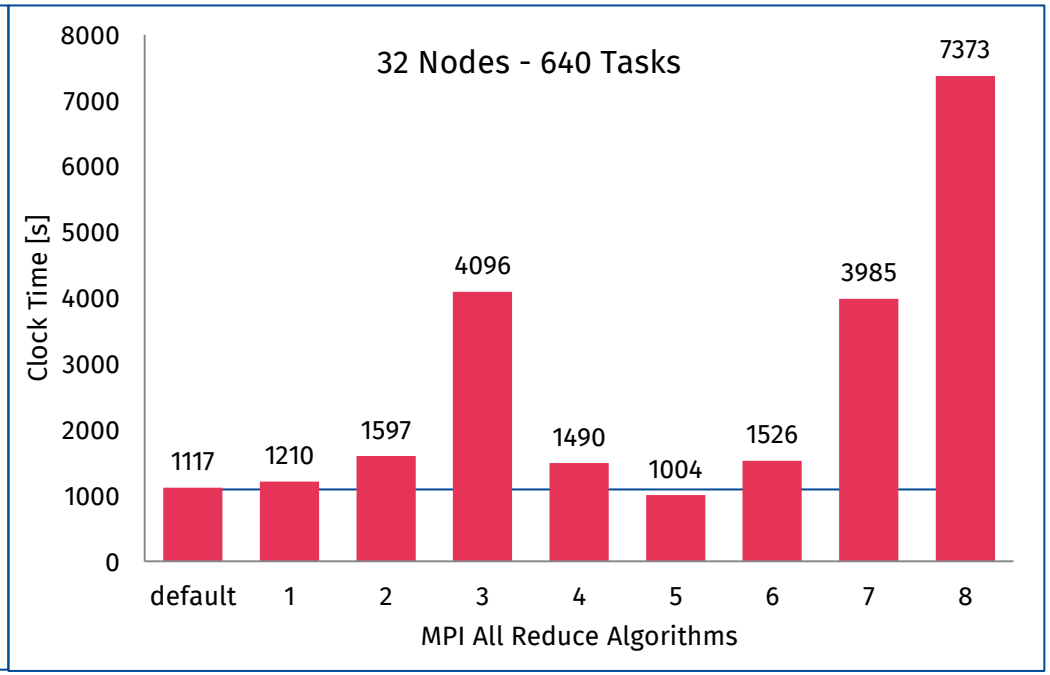
Dr. Oliver Schröder | o.schroeder@science-computing.de

MPI All Reduce Tuning (with Intel MPI)

Tests on SID B71010c (E5-2680v2)

science + computing
A Bull Group Company





- The best *All_Reduce* algorithm:
  - (5) Binominal Gather + Scatter
- Overall obtained improvement
  - None on 16 nodes – 320 Tasks
  - 11% on 32 Nodes – 640 Tasks

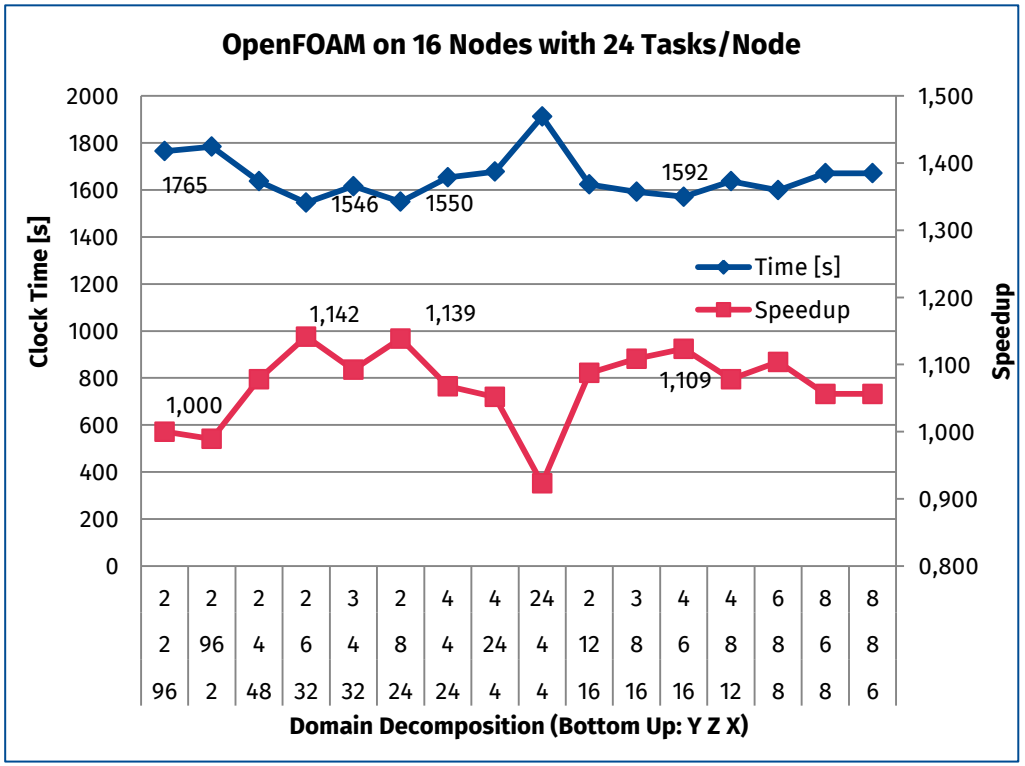
1. Recursive doubling algorithm
2. Rabenseifner's algorithm
3. Reduce + Bcast algorithm
4. Topology aware Reduce + Bcast algorithm
5. Binomial gather + scatter algorithm
6. Topology aware binominal gather + scatter algorithm
7. Shumilin's ring algorithm
8. Ring algorithm

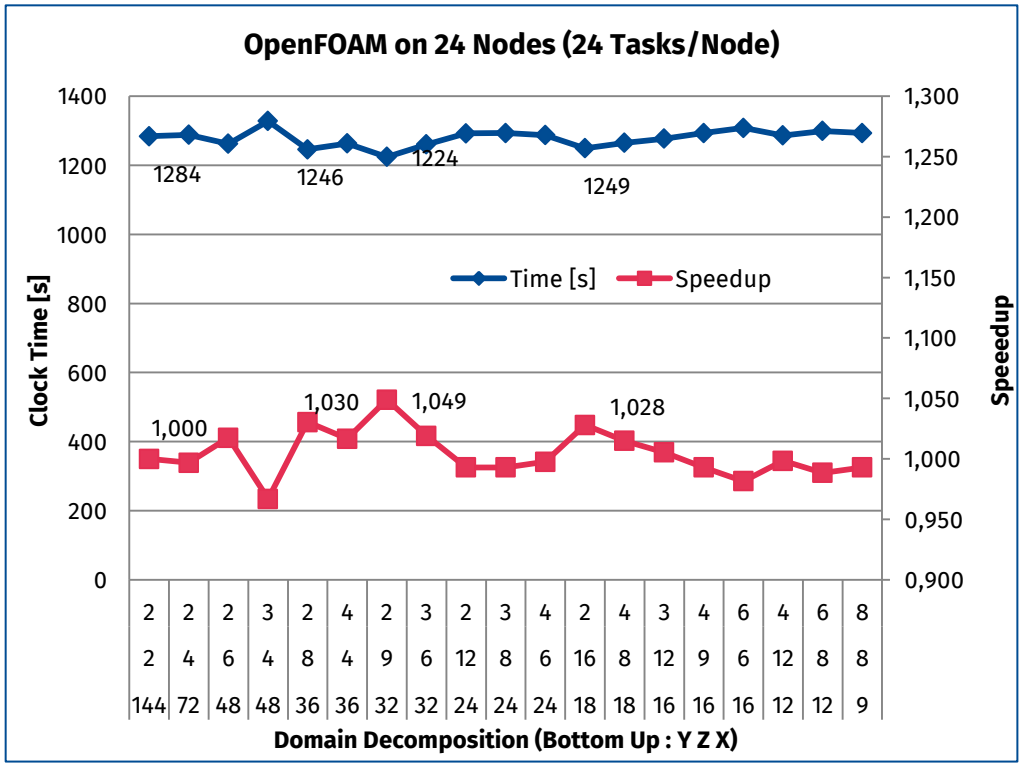
Dr. Oliver Schröder | o.schroeder@science-computing.de

Finding the Right Domain Decomposition

Tests on Robin with E6-2697v2 CPUs

science + computing
A Bull Group Company

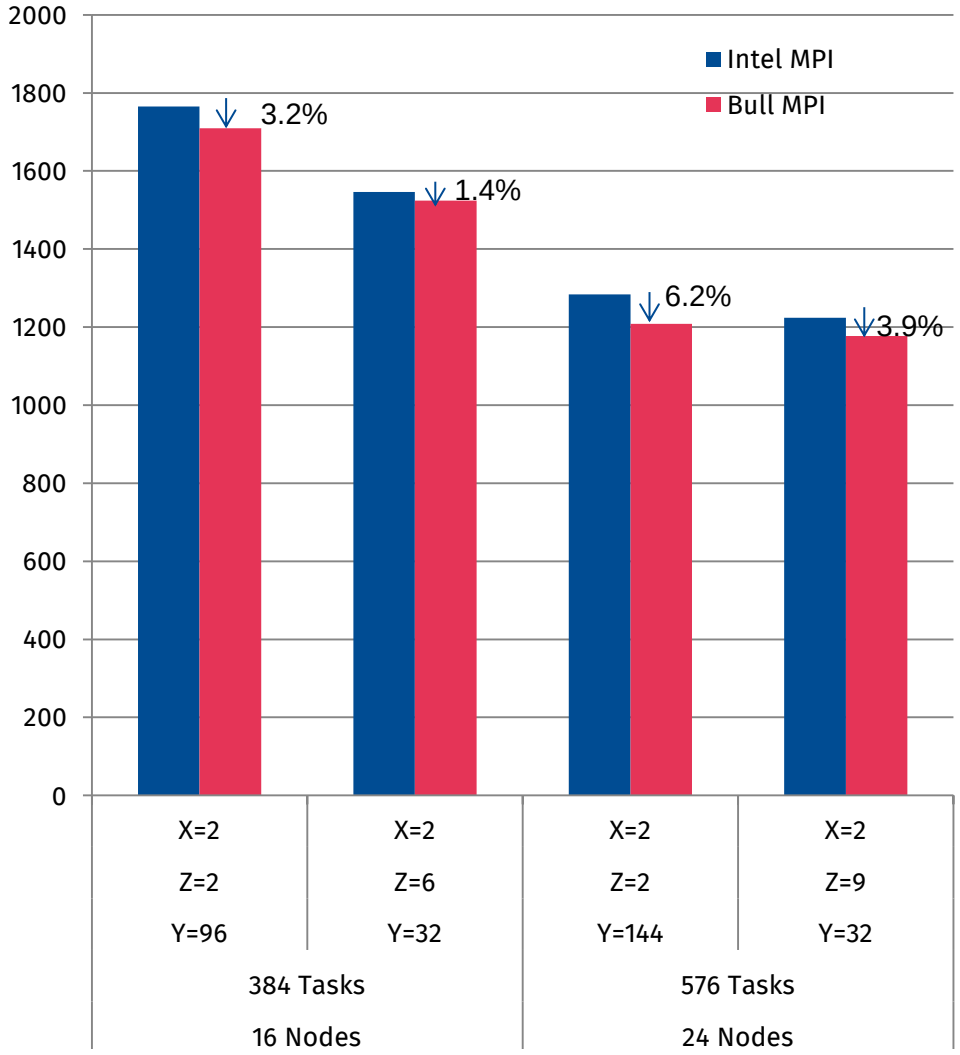




- We obtain
  - 10 - 14% improvement on 16 nodes
  - 3 – 5 % improvement on 24 nodes
- This means that decomposition *improves data locality* impacting thus the intra node communication, but it cannot have a high impact when the inter-node communication overhead increases.

Dr. Oliver Schröder | o.schroeder@science-computing.de

Bull MPI vs. Intel MPI
Tests on Robin with E6-2697v2 CPUs



- Bull MPI is based on OpenMPI
- It has been enhanced with many features such as:
  - effective abnormal pattern detection
  - network-aware collective operations
  - multi-path network failover.
- It has been designed to boost the performance of MPI applications thanks to full integration with
  - bullx PFS (based on Lustre)
  - bullx BM (based on SLURM)

Dr. Oliver Schröder | o.schroeder@science-computing.de

Conclusions

science + computing
A Bull Group Company

1. **CPU Frequency**
   - OpenFOAM shows a dependency under 50% on the CPU Frequency
2. **Cache Size**
   - OpenFOAM is dependent on the Cache Size per Core
3. **Limiting factor changes** as number of nodes change:
   1. *Memory Bandwidth*:
      OpenFOAM is memory bandwidth dependent on a *small number of nodes*
   2. *Communication:*
      MPI Time increases substantially with *increasing number of nodes*
4. Tuning **MPI All Reduce** Operations
   - Up to 11 % improvement can be obtained
5. **Domain Decomposition**
   - Finding the right mesh dimensions is very important
6. **File System**
   - We could *not* obtain *better results* with different stripe counts or size in Lustre

Dr. Oliver Schröder | o.schroeder@science-computing.de

science + computing

A Bull Group Company

Thank you for your attention.

Oliver Schröder
science + computing ag
www.science-computing.de

Email: o.schroeder@science-computing.de