

# First Experiences with Intel Cluster OpenMP

Christian Terboven, Dieter an Mey,

Dirk Schmidl, Marcus Wagner

*surname@rz.rwth-aachen.de*

Center for Computing and Communication  
RWTH Aachen University, Germany

# Agenda

- Intel Cluster OpenMP
  - OpenMP Memory Model
  - Consistency
- Micro-Benchmarks
  - EPCC
  - DSM investigations
- Applications
  - Jacobi
  - SMXV
  - GMRES
  - Panta
- Conclusion and Future Work

2

# Cluster OpenMP

- OpenMP versus MPI
  - Shared-Memory is more intuitive
  - OpenMP allows for incremental parallelization
  - BUT: Shared-Memory is bound to one machine
- OpenMP for clusters, based on ...
  - ... TreadMarks (subset of OpenMP only)
  - ... Omni/SCASH (no support for C++)
  - ... Linux kernel modifications, e.g. Kerrighed (compatibility)
- Intel Cluster OpenMP (CIOMP)
  - Full support of OpenMP 2.5 (no nesting support)
  - C, C++ and FORTRAN
  - First commercial implementation (based on TreadMarks)

3

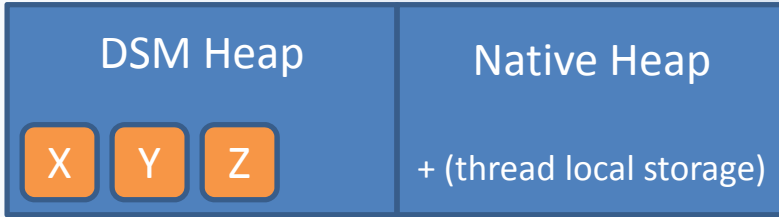
# Intel Cluster OpenMP

- OpenMP: Shared-Memory model
  - All threads share a common address space (shared memory)
  - Threads can have private data (explicit user control)
  - Fork-Join execution model
- Cluster OpenMP
  - Distributed-Shared-Memory Model (DSM)
  - Intel: `sharable` (not all variables can be made sharable automatically) + `kmp_sharable_malloc()`
- Weak memory model
  - Temporary View: Memory consistency is guaranteed only after certain points, namely implicit and explicit flushes

4

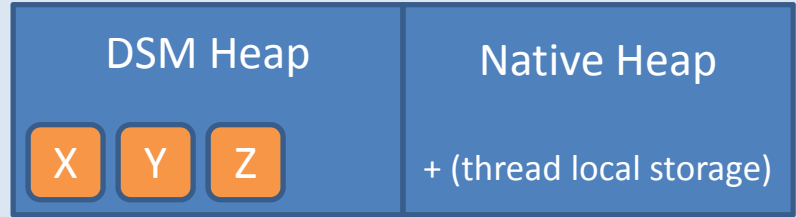
# Consistency model in DSM

Node 0: ClOMP process A w/ 1 thread

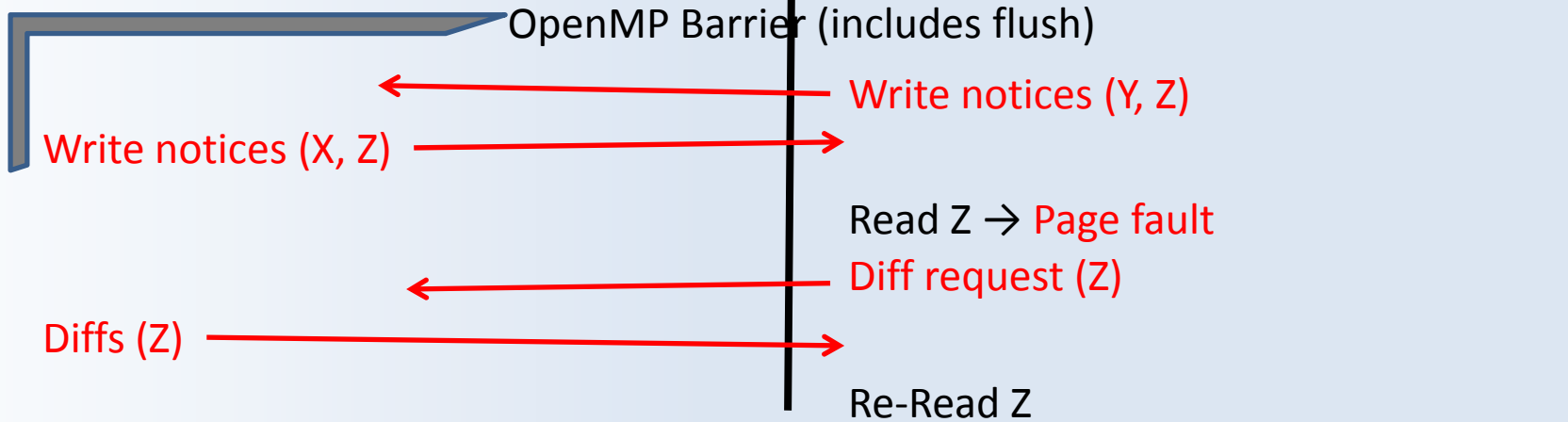


Write X → Twin page creation (X)  
Write Z → Twin page creation (Z)

Node 1: ClOMP process B w/ 1 thread



Write Y → Twin page creation (Y)  
Write Z → Twin page creation (Z)



- Initially, pages in the DSM are read + write protected
- Write notices are sent to node 0 and then propagated

5

# Agenda

- Intel Cluster OpenMP
  - OpenMP Memory Model
  - Consistency
- Micro-Benchmarks
  - EPCC
  - DSM investigations
- Applications
  - Jacobi
  - SMXV
  - GMRES
  - Panta
- Conclusion and Future Work

6

# EPCC Micro-Benchmarks on two nodes

J. M. Bull. Measuring Synchronization and Scheduling Overheads in OpenMP. 1999.

	OpenMP	CIOMP (Eth)	CIOMP (IB)
PARALLEL FOR			
1 thread p. node	0.31	482.35	464.38
2 threads p. node	1.00	925.54	662.42
4 threads p. node	1.12	1004.62	776.56
BARRIER			
1 thread p. node	0.01	482.14	464.08
2 threads p. node	0.43	648.14	528.67
4 threads p. node	0.60	689.81	631.24
REDUCTION			
1 thread p. node	0.35	482.17	464.59
2 threads p. node	1.54	1249.04	817.48
4 threads p. node	2.32	2084.89	1195.40

**Overhead in microseconds [us].** Intel 10.1.011 compilers for 64bit Linux.

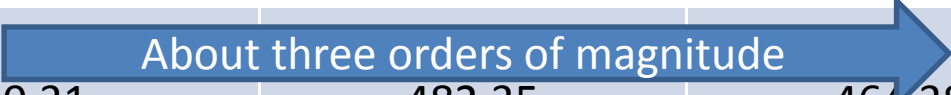
Gigabit Ethernet (Eth): Dell PE 1950 cluster, 2x Intel Xeon 5160 (dual-core, 3.0 GHz).

4x DDR InfiniBand (IB): FSC RX 200 cluster, 2x Intel Xeon 5450 (quad-core, 3.0 GHz).

7

# EPCC Micro-Benchmarks on two nodes

J. M. Bull. Measuring Synchronization and Scheduling Overheads in OpenMP. 1999.

	OpenMP	CIOMP (Eth)	CIOMP (IB)
PARALLEL FOR	About three orders of magnitude 		
1 thread p. node	0.31	482.35	464.38
2 threads p. node	1.00	925.54	662.42
4 threads p. node	1.12	1004.62	776.56
BARRIER			
1 thread p. node	0.01	482.14	464.08
2 threads p. node	0.43	648.14	528.67
4 threads p. node	0.60	689.81	631.24
REDUCTION			
1 thread p. node	0.35	482.17	464.59
2 threads p. node	1.54	1249.04	817.48
4 threads p. node	2.32	2084.89	1195.40

**Overhead in microseconds [us].** Intel 10.1.011 compilers for 64bit Linux.

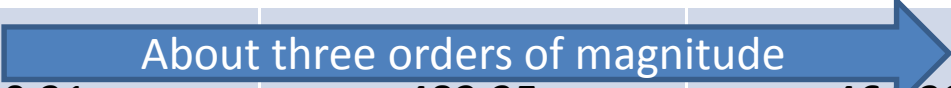

Gigabit Ethernet (Eth): Dell PE 1950 cluster, 2x Intel Xeon 5160 (dual-core, 3.0 GHz).

4x DDR InfiniBand (IB): FSC RX 200 cluster, 2x Intel Xeon 5450 (quad-core, 3.0 GHz).

8

# EPCC Micro-Benchmarks on two nodes

J. M. Bull. Measuring Synchronization and Scheduling Overheads in OpenMP. 1999.

	OpenMP	CIOMP (Eth)	CIOMP (IB)
PARALLEL FOR	About three orders of magnitude 		
1 thread p. node	0.31	482.35	464.38
2 threads p. node	1.00	925.54	662.42
4 threads p. node	1.12	1004.62	776.56
BARRIER			
1 thread p. node	0.01	482.14	464.08
2 threads p. node	0.43	648.14	528.67
4 threads p. node	0.60	689.81	631.24
REDUCTION		IB is significantly faster 	
1 thread p. node	0.35	482.17	464.59
2 threads p. node	1.54	1249.04	817.48
4 threads p. node	2.32	2084.89	1195.40

**Overhead in microseconds [us].** Intel 10.1.011 compilers for 64bit Linux.

Gigabit Ethernet (Eth): Dell PE 1950 cluster, 2x Intel Xeon 5160 (dual-core, 3.0 GHz).

4x DDR InfiniBand (IB): FSC RX 200 cluster, 2x Intel Xeon 5450 (quad-core, 3.0 GHz).



## DSM investigations with two threads

	Allocate page in DSM heap	Read page from other CIOMP thread	Write page to other CIOMP thread
OpenMP	0.85	1.8	2.32
CIOMP (Eth)			
1 node	3.88	1.76	2.46
2 nodes	12.82	242.87	236.06
CIOMP (IB)			
1 node	4.37	1.78	2.49
2 nodes	22.96	94.94	94.38

**Overhead in microseconds [us].**

- All measurements done with two threads:
  - OpenMP: two threads on one node
  - Cluster OpenMP: two threads in total on one / two nodes

10

## DSM investigations with two threads

	Allocate page in DSM heap	Read page from other CIOMP thread	Write page to other CIOMP thread
OpenMP	0.85	1.8	2.32
CIOMP (Eth)	↑ Eth is faster		
1 node		3.88	1.76
2 nodes	12.82	242.87	236.06
CIOMP (IB)			
1 node	4.37	1.78	2.49
2 nodes	22.96	94.94	94.38

**Overhead in microseconds [us].**

- All measurements done with two threads:
  - OpenMP: two threads on one node
  - Cluster OpenMP: two threads in total on one / two nodes

# DSM investigations with two threads

	Allocate page in DSM heap	Read page from other CIOMP thread	Write page to other CIOMP thread
OpenMP	0.85	1.8	2.32
CIOMP (Eth)			
1 node	3.88	1.76	2.46
2 nodes	12.82	242.87	236.06
CIOMP (IB)			
1 node	4.37	1.78	2.49
2 nodes	22.96	94.94	94.38

Eth is faster

IB is faster

1.78  
94.94

**Overhead in microseconds [us].**

- All measurements done with two threads
  - OpenMP: two threads on one node
  - Cluster OpenMP: two threads in total

Starting Intel Cluster OpenMP with multiple threads per process (node) can improve performance significantly.

# Agenda

- Intel Cluster OpenMP
  - OpenMP Memory Model
  - Consistency
- Micro-Benchmarks
  - EPCC
  - DSM investigations
- Applications
  - Jacobi
  - SMXV
  - GMRES
  - Panta
- Conclusion and Future Work

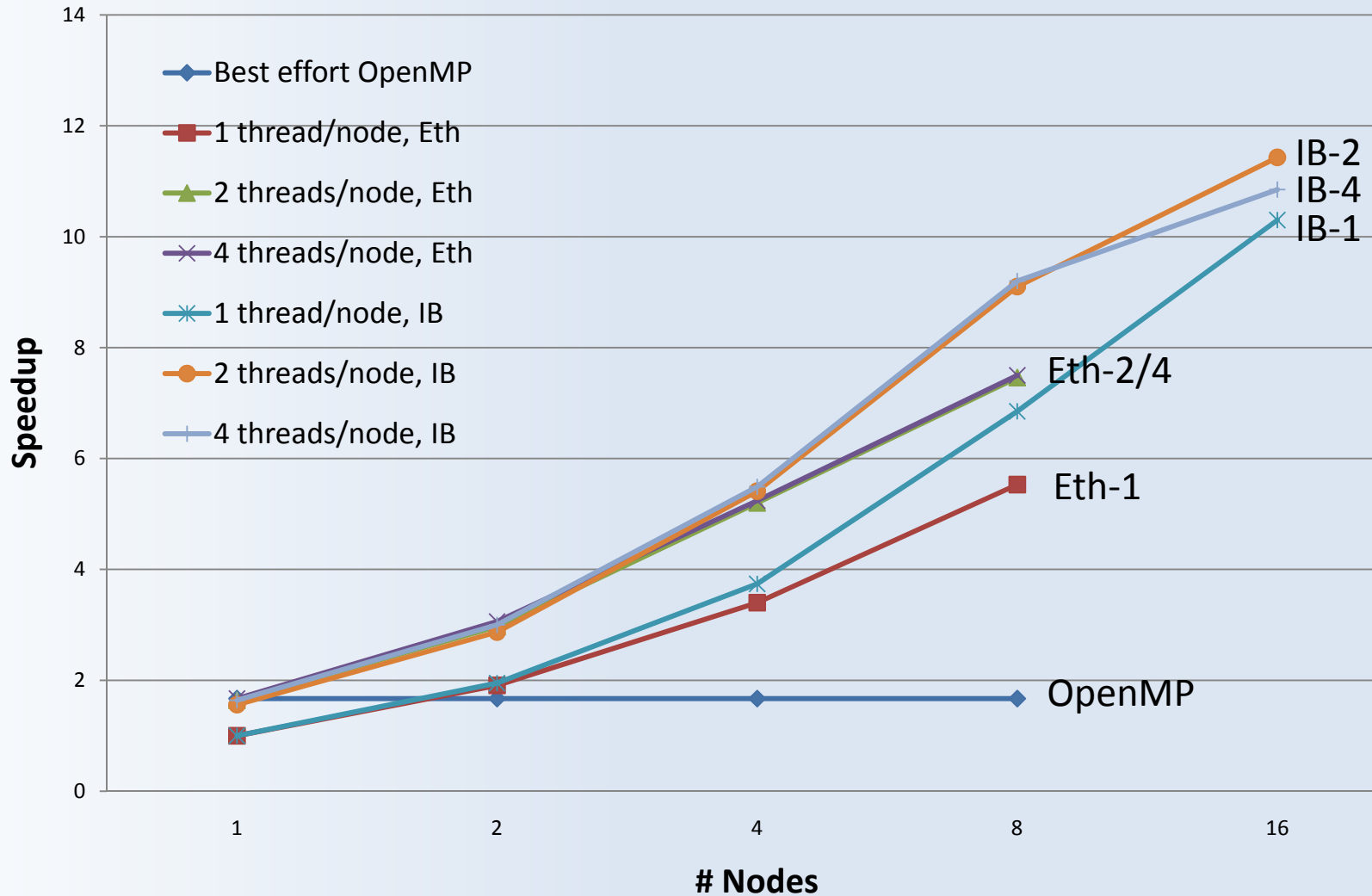
13

## Jacobi

- Simple example program: Jacobian solver ([www.openmp.org](http://www.openmp.org))
  - Matrix size of 6000 x 6000
  - Limited to 100 iterations
  - Parallelization strategy: Domain decomposition
- Performance tuning:
  - Thread binding (`KMP_AFFINITY`)
  - Strategy: *scattered*

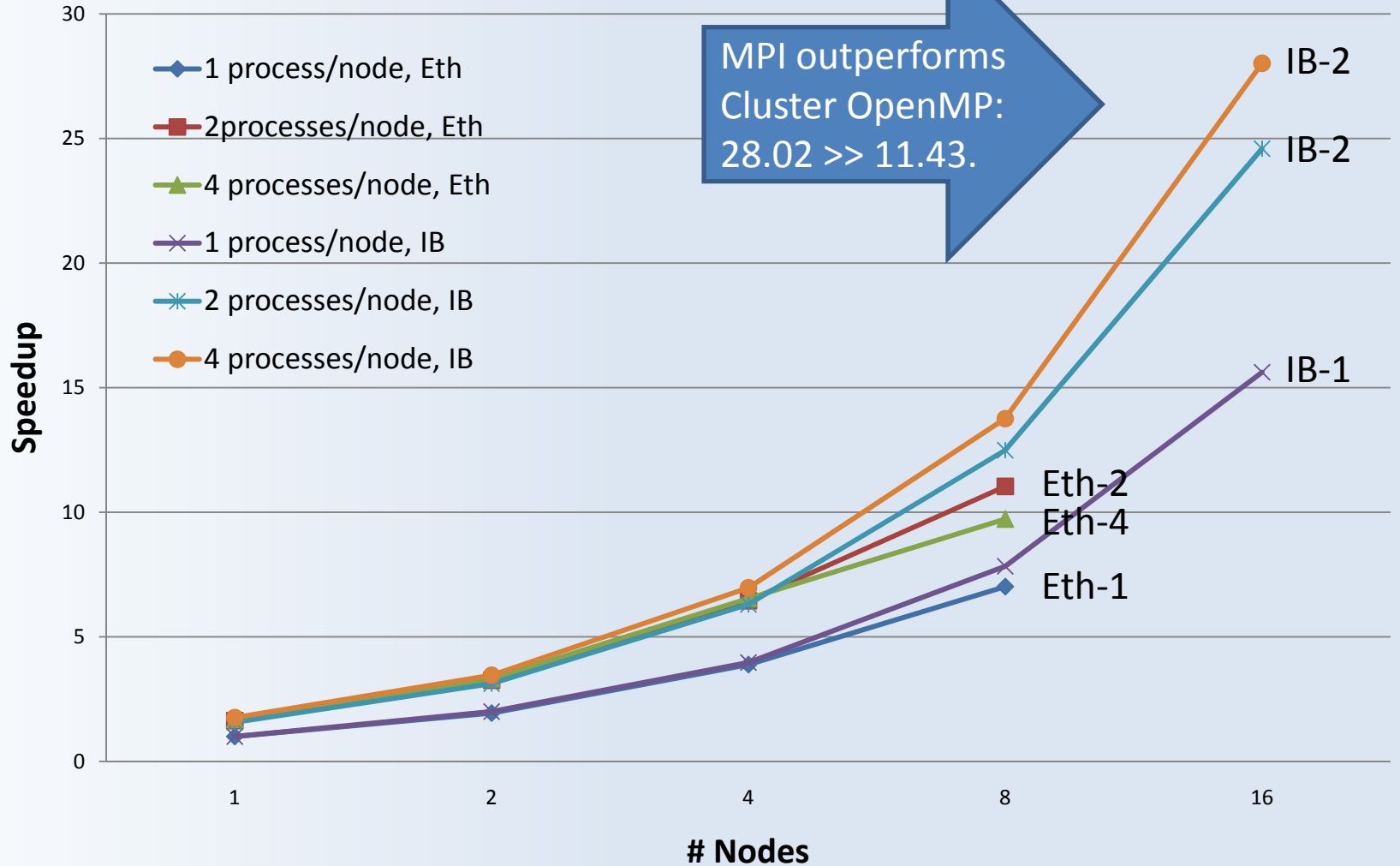
14

# Jacobi: Intel Cluster OpenMP



15

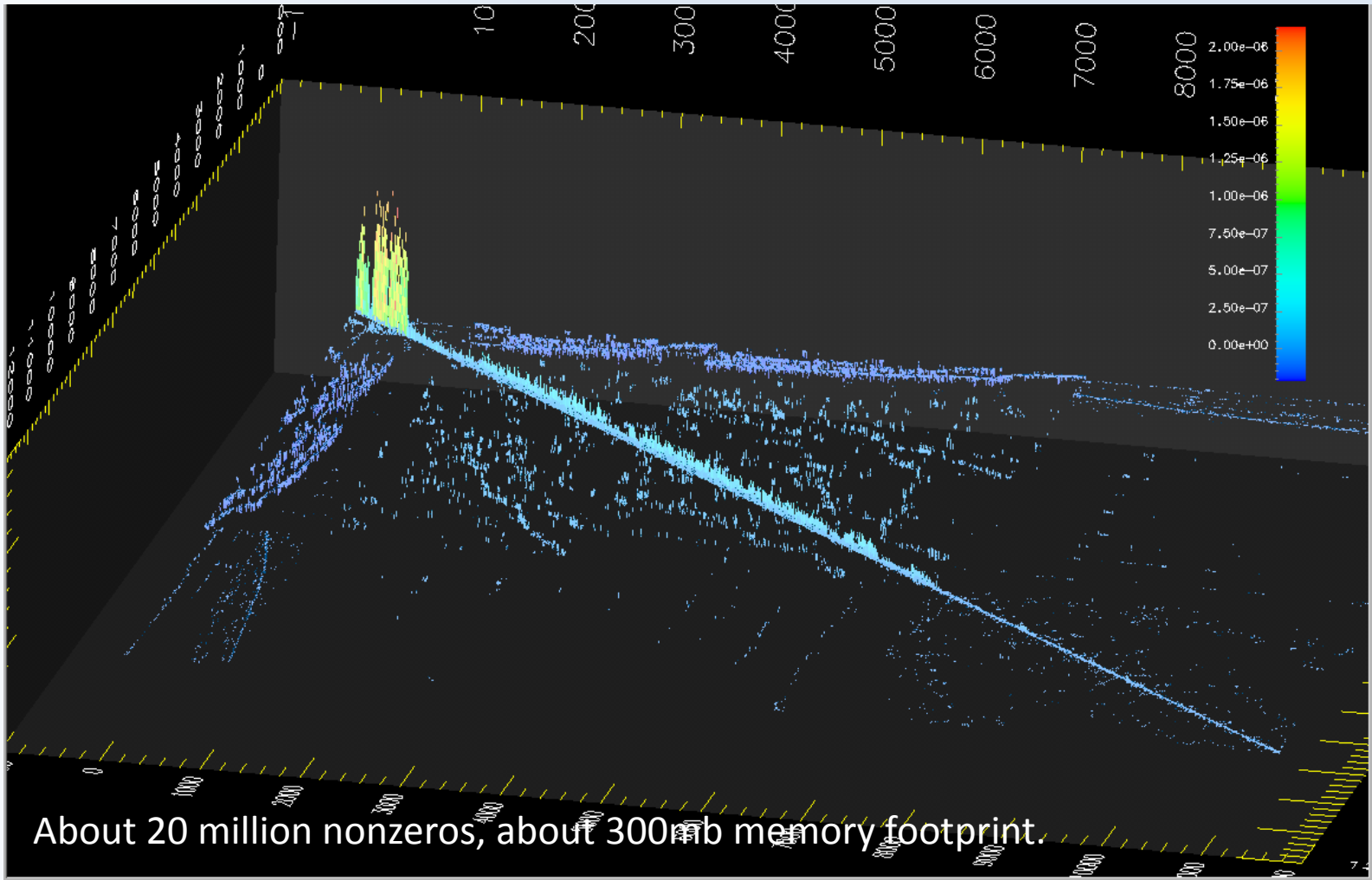
# Jacobi: sync. MPI



16

# Sparse Matrix-Vector-Multiplication

- DROPS, a C++ Navier-Stokes solver of two-phase flows



17

Center for

Computing and  
Communication

Cluster OpenMP

Micro-  
Benchmarks

Applications

Conclusion &  
Future Work

# Sparse Matrix-Vector-Multiplication [Mflop/s]

	<u>rows</u>		<u>nonzeros</u>	
	1 thread p. node	8 threads p. node	1 thread p. node	8 threads p. node
OpenMP, UMA	502.9	976.5	501.6	998.1
OpenMP, ccNUMA	326.3	793.9	324.5	1147.6
CIOMP, Eth, 1 node	548.0	887.2	551.8	939.4
CIOMP, Eth, 2 nodes	113.0	540.1	1058.7	1382.4
CIOMP, Eth, 4 nodes	14.5	136.8	2037.9	2435.6
CIOMP, IB, 1 node	502.01	812.15	503.61	914.43
CIOMP, IB, 2 nodes	365.87	617.22	994.25	1582.62
CIOMP, IB, 4 nodes	290.28	424.88	1923.08	2754.78
CIOMP, IB, 8 nodes	271.51	318.15	3572.34	4648.44
CIOMP, IB, 16 nodes	281.67	246.08	5989.76	7228.97

ccNUMA: Sun Fire V40z server, 4x AMD Opteron 848 (single-core, 2.2 GHz).

- rows-strategy: parallel loop over #rows, dynamic loop sched.
- nonzeros-strategy: #nonzeros statically partitioned

# Sparse Matrix-Vector-Multiplication [Mflop/s]

	<u>rows</u>		<u>nonzeros</u>	
	1 thread p. node	8 threads p. node	1 thread p. node	8 threads p. node
OpenM	502.9	976.5	501.6	998.1
OpenM MA	326.3	793.9	324.5	1147.6
CIOMP node	548.0	887.2	551.8	939.4
CIOMP nodes	113.0	540.1	1058.7	1382.4
CIOMP nodes	14.5	136.8	2037.9	2435.6
CIOMP	502.01	812.15	503.61	914.43
CIOMP nodes	365.87	617.22	994.25	1582.62
CIOMP, 16 nodes	290.28	424.88	1923.08	2754.78
CIOMP, IB, 8 nodes	271.51	318.15	3572.34	4648.44
CIOMP, IB, 16 nodes	281.67	246.08	5989.76	7228.97

Cluster OpenMP similar to ccNUMA

ccNUMA: Sun Fire V40z server, 4x AMD Opteron 848 (single-core, 2.2 GHz).

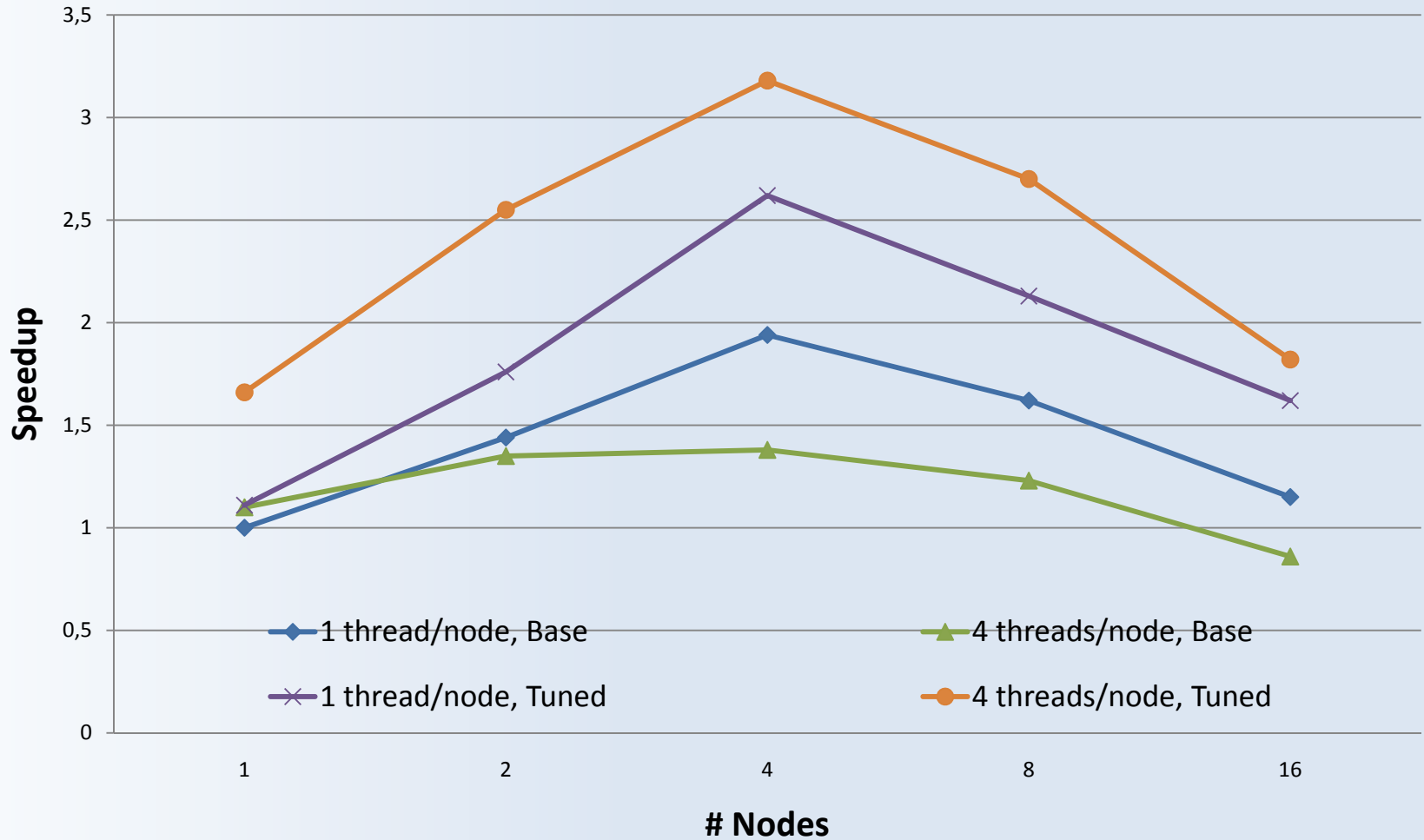
- rows-strategy: parallel loop over #rows, dynamic loop sched.
- nonzeros-strategy: #nonzeros statically partitioned

## GMRES

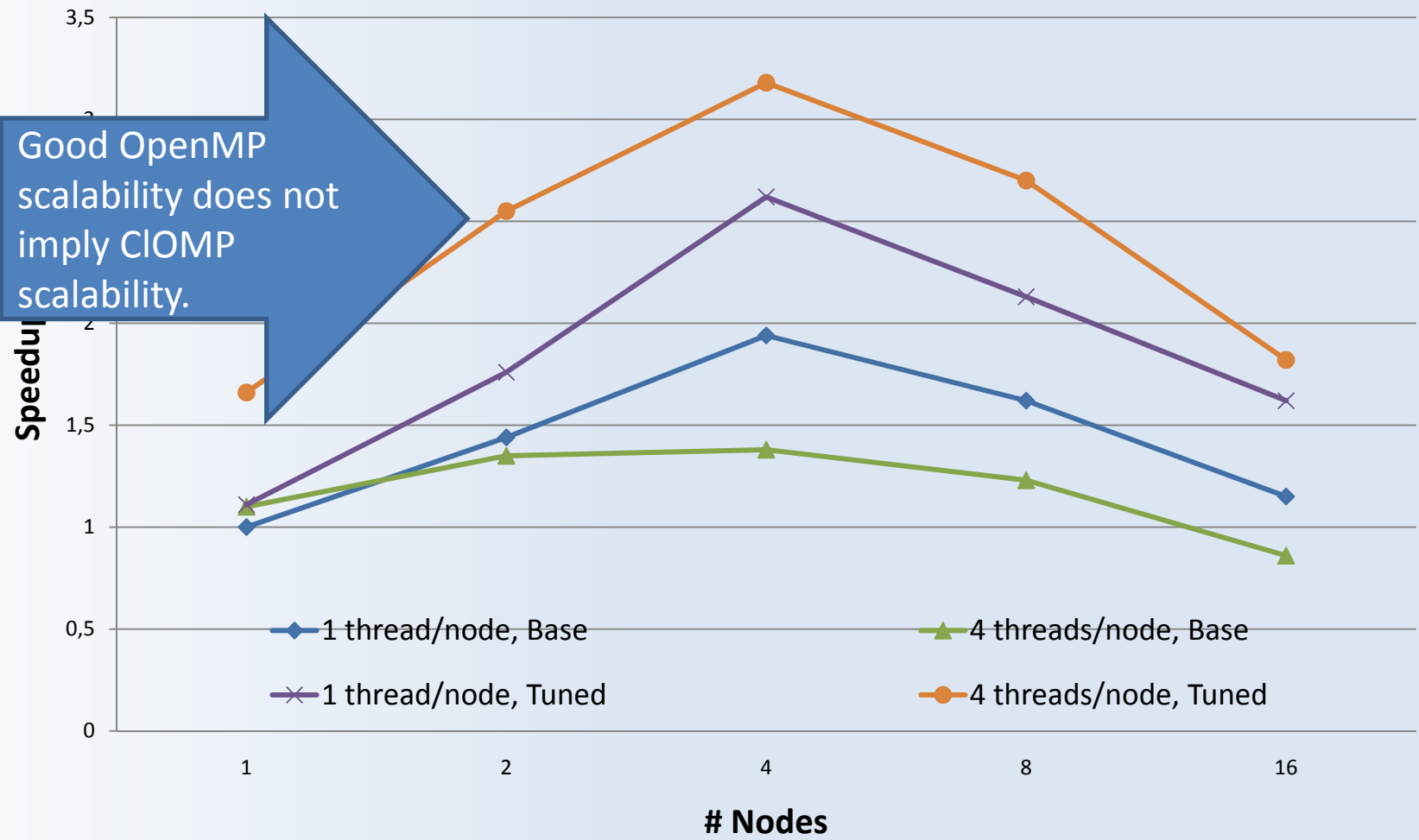
- GMRES Solver kernel from DROPS
  - Written in C++
  - Same matrix structure as shown above
- Comparing two versions
  - Base: Original OpenMP parallelization
  - Tuned: Parallelization tuned for Cluster OpenMP

20

# GMRES

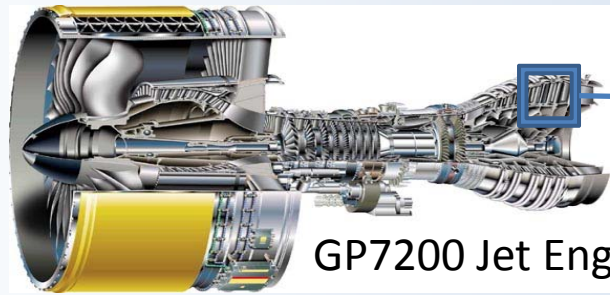


# GMRES

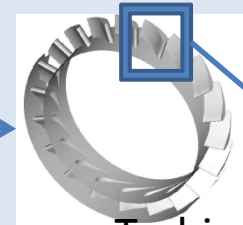


# PANTA

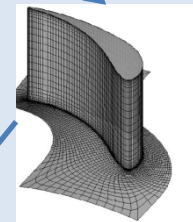
- o Developed at IST at RWTH Aachen University
  - Computation of turbomachinery flow → solution of PDEs
  - About 50,000 lines of Fortran90 code
  - Hybrid code. Here: OpenMP-parallel loop over 80 inv. zones



GP7200 Jet Engine for Airbus A380

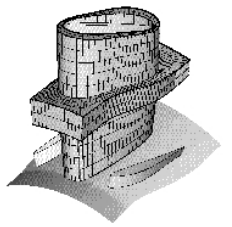


Low Pressure Turbine Blade Row

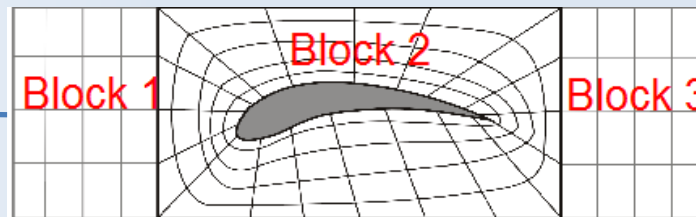


Low Pressure Turbine Blade Channel

Alternating Direction Inversion Zones

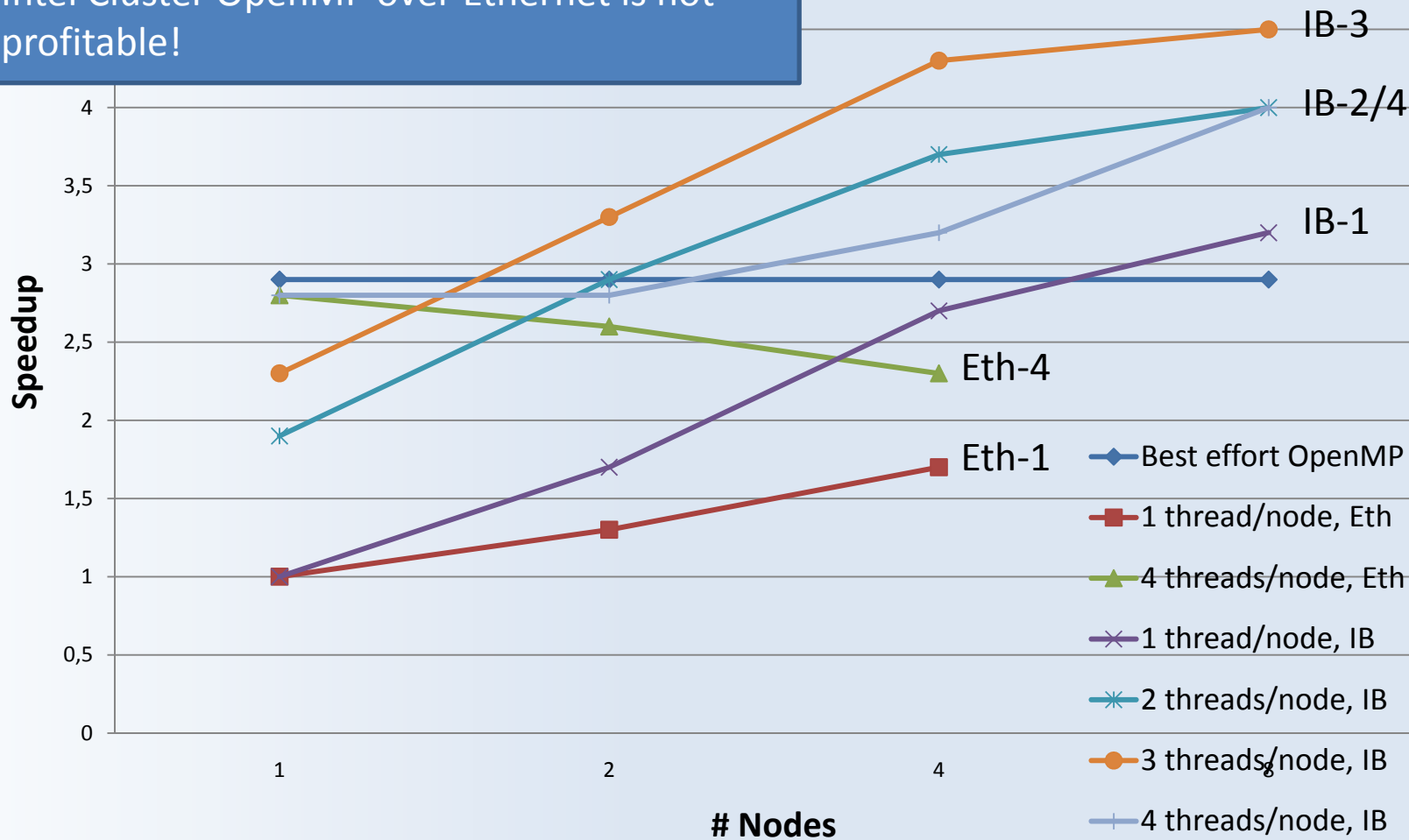


Blocks



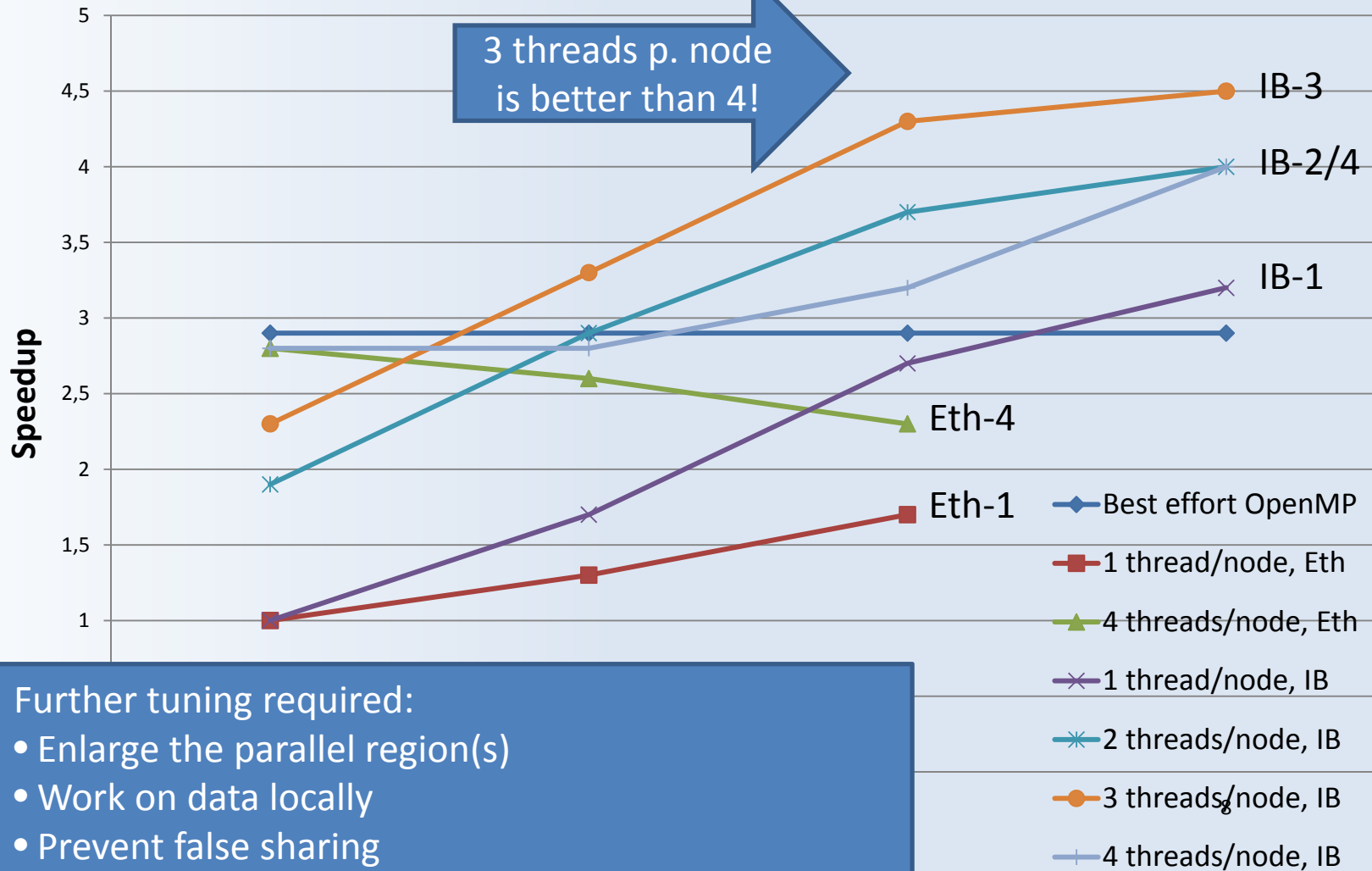
# PANTA

OpenMP scalability on one node is limited.  
Intel Cluster OpenMP over Ethernet is not profitable!



24

# PANTA



25

# Agenda

- Intel Cluster OpenMP
  - OpenMP Memory Model
  - Consistency
- Micro-Benchmarks
  - EPCC
  - DSM investigations
- Applications
  - Jacobi
  - SMXV
  - GMRES
  - Panta
- Conclusion and Future Work

26

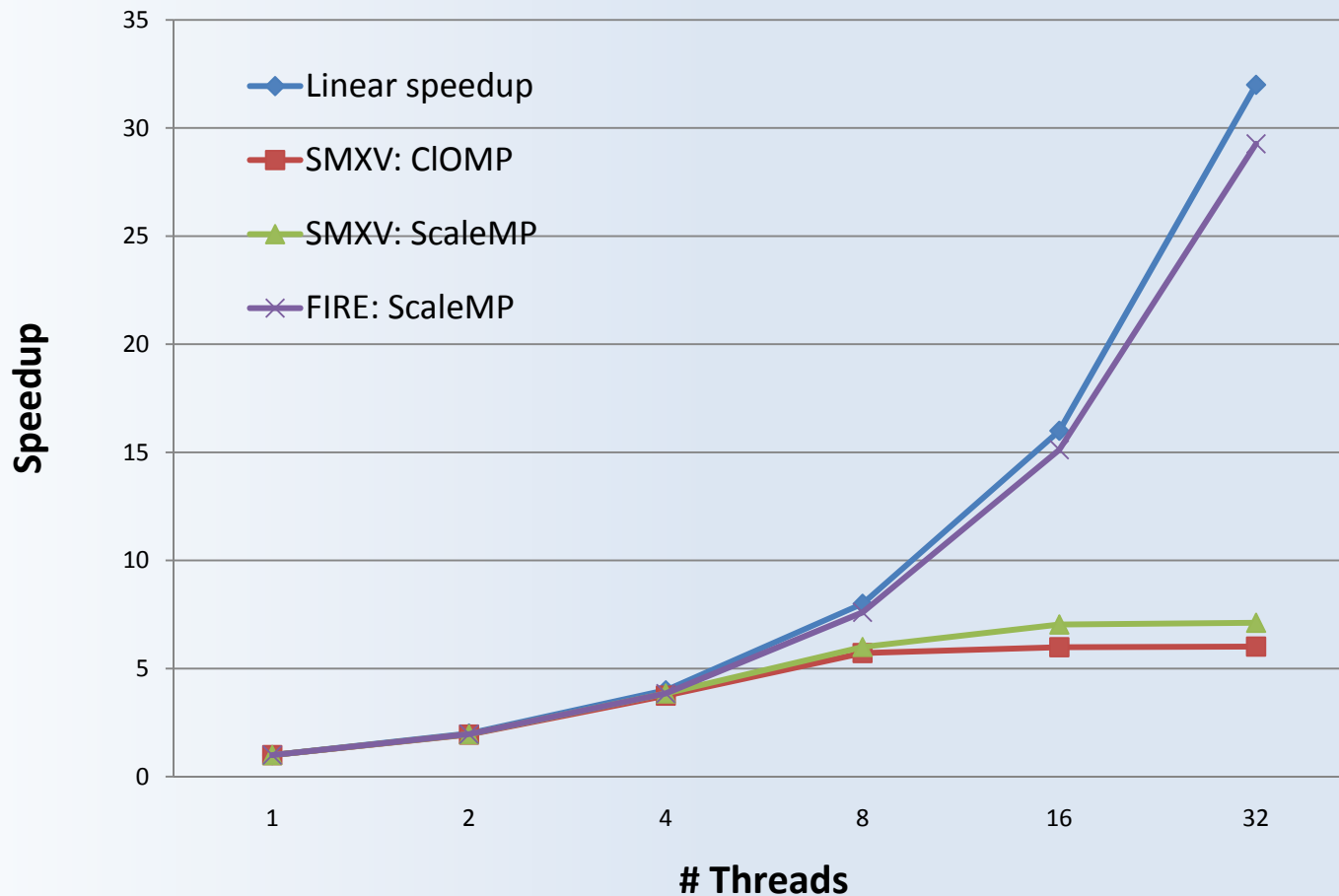
## Conclusion and Future Work (1/2)

- Cluster OpenMP brings OpenMP onto a cluster
  - Takes advantage of relaxed consistency model
  - OpenMP primitives become significantly more expensive
- Intel Cluster OpenMP
  - Proved to be successful for several small applications
  - A fast network (IB) is crucial for application performance
  - Problems with C++ programs employing the STL
  - We had no significant success with a real application yet
- Future Work
  - Devise tuning strategies learned from GMRES
  - Multi-level parallelism (PThreads, Threading Building Blocks)
  - We (still) believe that Distributed-Shared-Memory is well-suited for many (of our) applications ...

27

## Conclusion and Future Work (2/2)

- OpenMP on ScaleMP: Single-system-image over InfiniBand
  - Nearly linear speedup with FIRE, better than CIOMP for SMXV



28

Center for

Computing and

Communication

Cluster OpenMP

Micro-  
Benchmarks

Applications

Conclusion &  
Future Work

End

Thank you  
for your  
attention!

29