Optimizing a Conjugate Gradient Solver with Non-Blocking Collective Operations

T. Hoefler, P. Gottschling, W. Rehm, A. Lumsdaine

Open Systems Lab
Indiana University
Bloomington, USA

Computer Architecture Group
Technical University of Chemnitz
Chemnitz, Germany

EuroPVM/MPI'06 - ParSim'06 Special Session
Bonn, Germany
20th September 2006

Non-Blocking Collectives - Why?

- combine advantages of collective operations and overlapping
- enable use of hardware parallelism (overlap)
- $\Rightarrow$ latency is hidden with bandwidth
- additional new interesting features (non-blocking barrier)
- pseudo-synchronization in the background
- tolerate parallel process skew

OS Noise and Process Skew?

Iskra et. al. (2006) *"The Influence of Operating Systems on the Performance of Collective Operations at Extreme Scale"*

Non-Blocking Collectives - Why?

- combine advantages of collective operations and overlapping
- enable use of hardware parallelism (overlap)
- $\Rightarrow$ latency is hidden with bandwidth
- additional new interesting features (non-blocking barrier)
- pseudo-synchronization in the background
- tolerate parallel process skew

OS Noise and Process Skew?

Iskra et. al. (2006) *"The Influence of Operating Systems on the Performance of Collective Operations at Extreme Scale"*

Process Skew

- caused by OS interference or unbalanced application
- especially if processors are overloaded
- worse for big systems
- can cause dramatic performance decrease
- all nodes wait for the last

Does it really matter?

Petrini et. al. (2003) *"The Case of the Missing Supercomputer Performance: Achieving Optimal Performance on the 8,192 Processors of ASCI Q"*

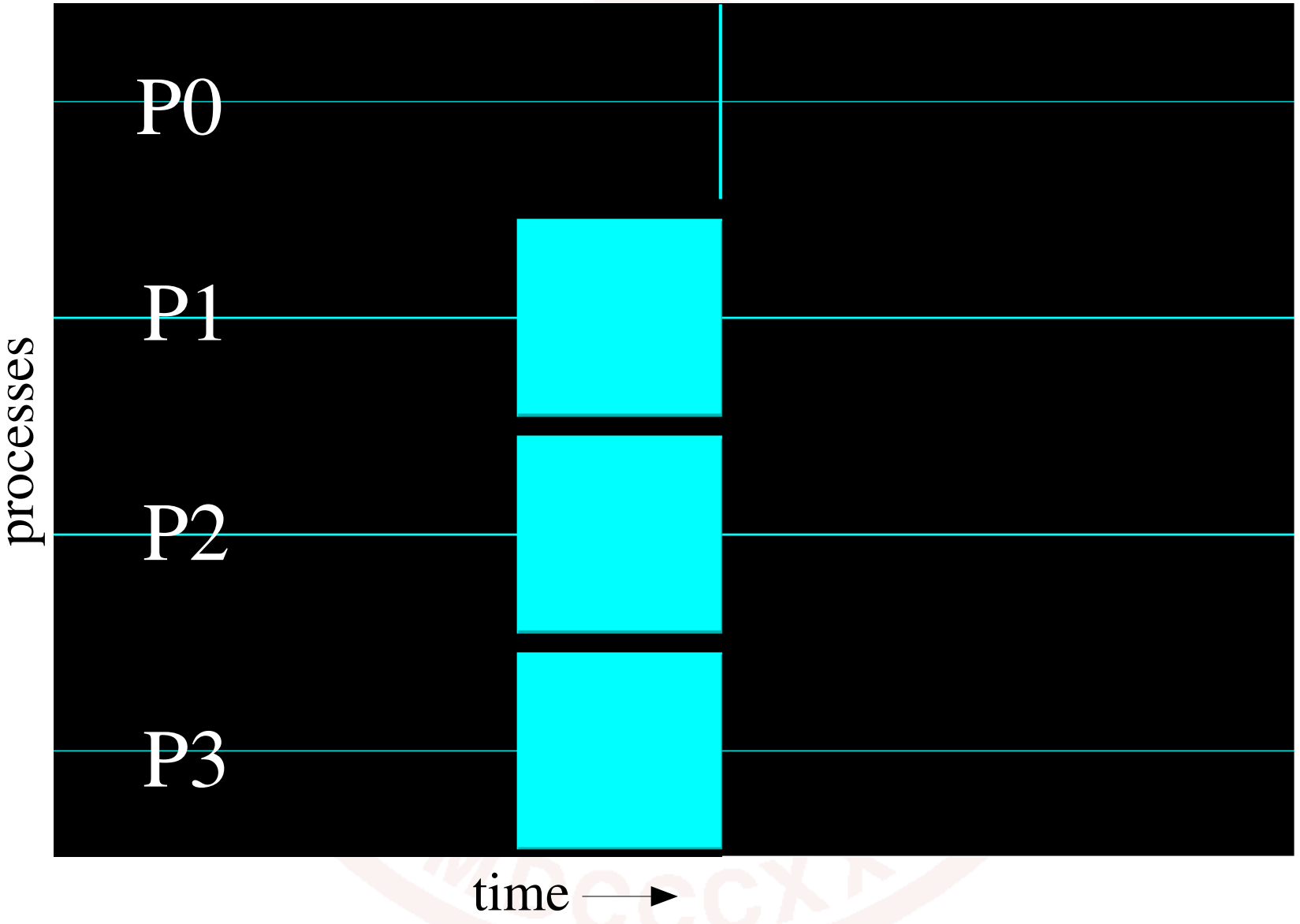
Process Skew

- caused by OS interference or unbalanced application
- especially if processors are overloaded
- worse for big systems
- can cause dramatic performance decrease
- all nodes wait for the last

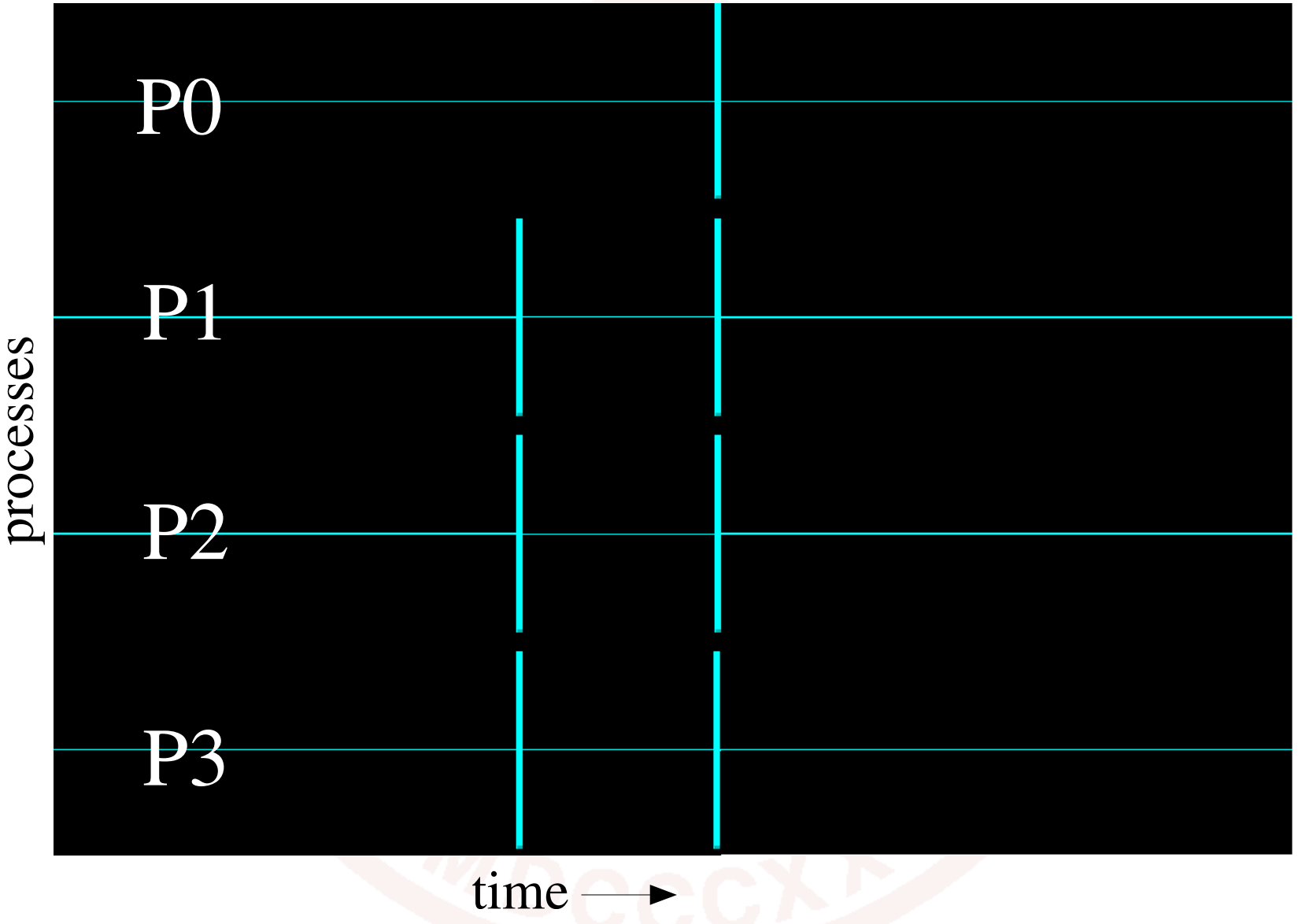
Does it really matter?

Petrini et. al. (2003) *"The Case of the Missing Supercomputer Performance: Achieving Optimal Performance on the 8,192 Processors of ASCI Q"*

Process Skew - MPI Example - Jumpshot



Process Skew - NBC Example - Jumpshot



Non-Blocking Collectives - Interface

- extension to MPI-2
- ”mixture” between non-blocking ptp and collectives
- uses MPI_Requests and MPI_Test/MPI_Wait

```
MPI_Ibcast(buf1, p, MPI_INT, 0, MPI_COMM_WORLD, &req);
MPI_Wait(&req);
```

Standard Proposal

Hoefler et. al. (2006): *”Non-Blocking Collective Operations for MPI-2”*

Non-Blocking Collectives - Interface

- extension to MPI-2
- ”mixture” between non-blocking ptp and collectives
- uses MPI_Requests and MPI_Test/MPI_Wait

```
MPI_Ibcast(buf1, p, MPI_INT, 0, MPI_COMM_WORLD, &req);
MPI_Wait(&req);
```

Standard Proposal

Hoefler et. al. (2006): *”Non-Blocking Collective Operations for MPI-2”*

Non-Blocking Collectives - Implementation

- implementation available with LibNBC
- written in ANSI-C and uses only MPI-1
- central element: collective schedule
- a coll-algorithm can be represented as a schedule

Example: dissemination barrier, 4 nodes, node 0:

send to 1	recv from 3	end	send to 2	recv from 2	end

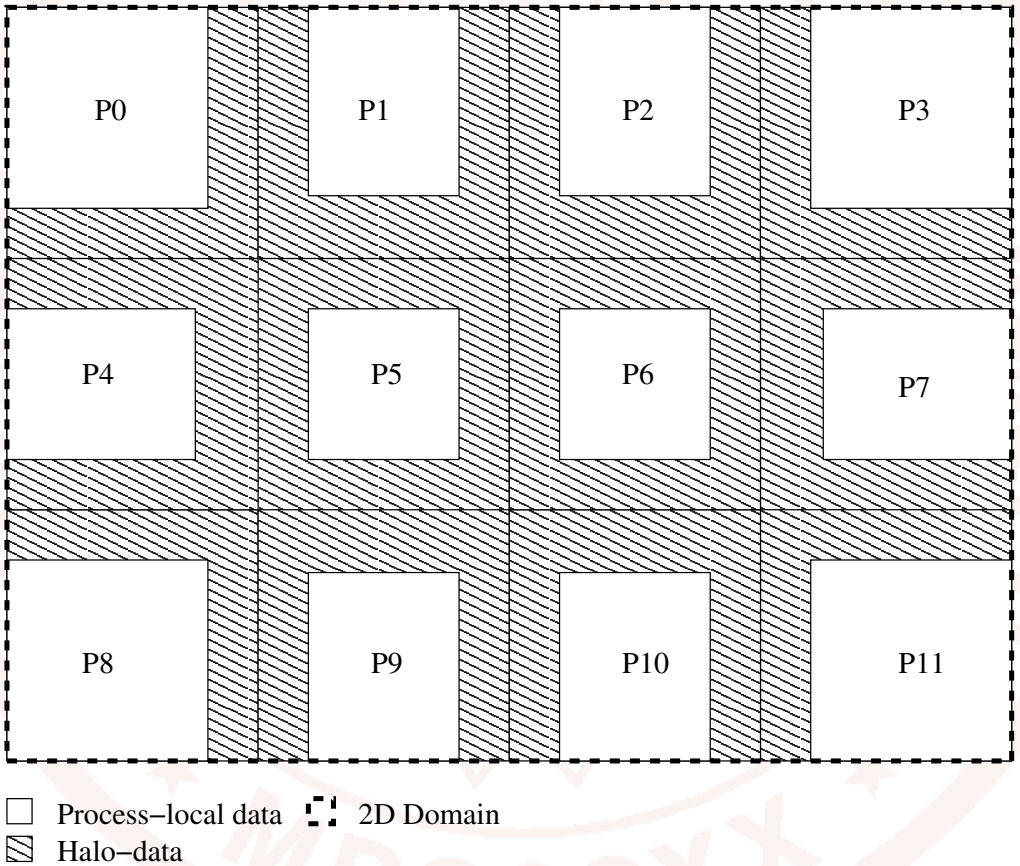
LibNBC download: `http://www.unixer.de/NBC`

Linear Solvers - Domain Decomposition

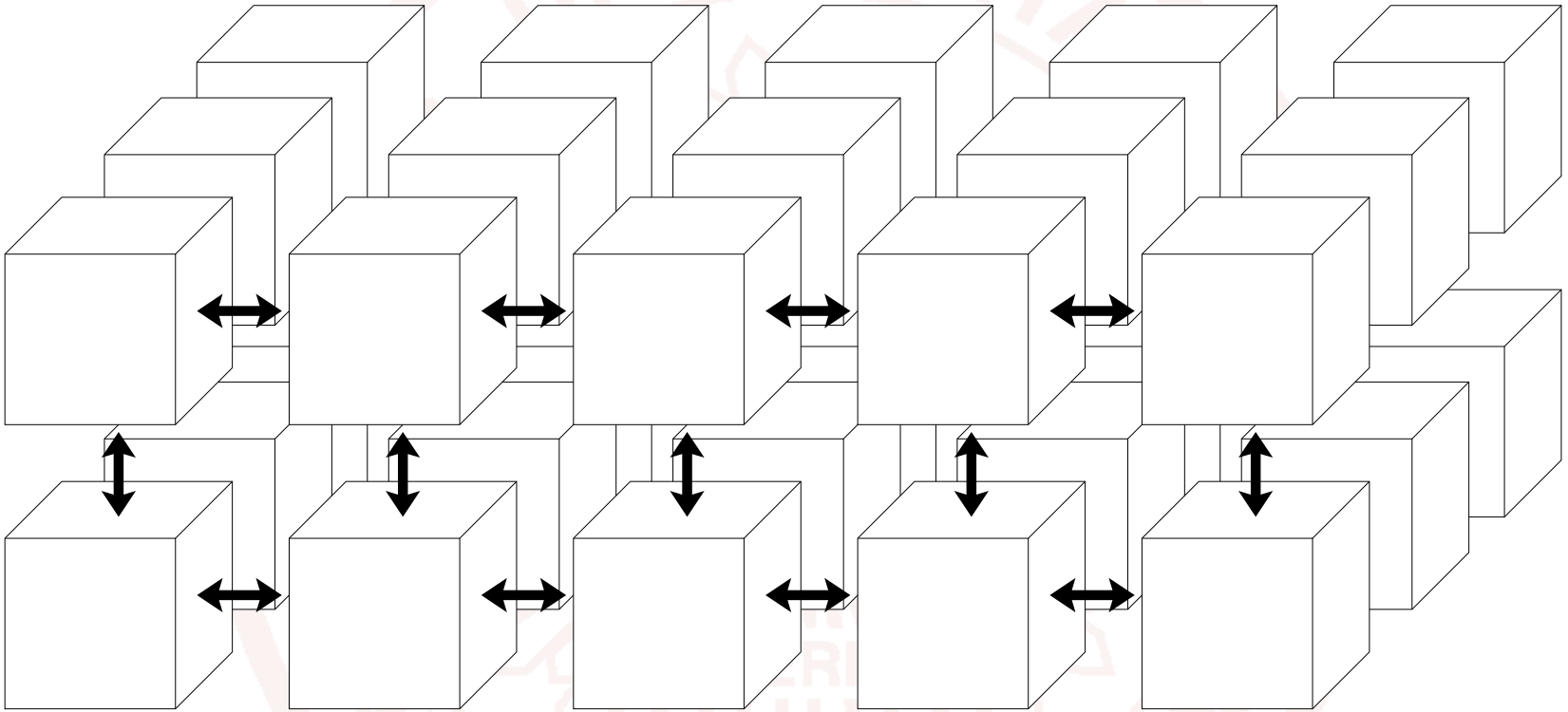
- iterative linear solvers are used in many scientific kernels
- often used operation is vector-matrix-multiply
- matrix is domain-decomposed (e.g., 3D)
- only outer (border) elements need to be communicated
- can be overlapped

Domain Decomposition

- nearest neighbor communication
- can be implemented with MPI_Alltoallv



Communication 3D

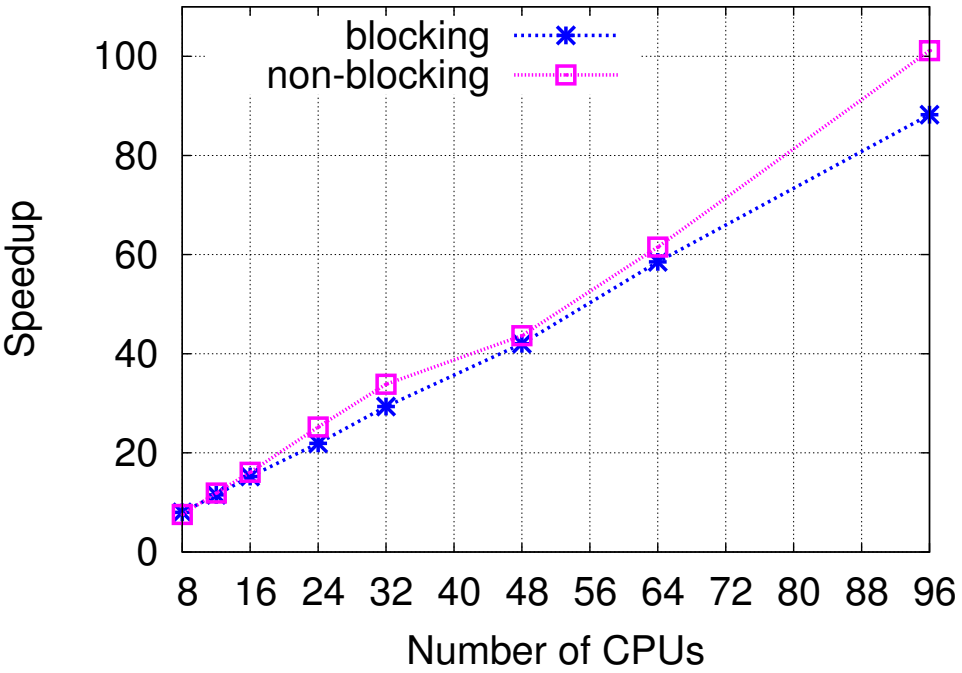


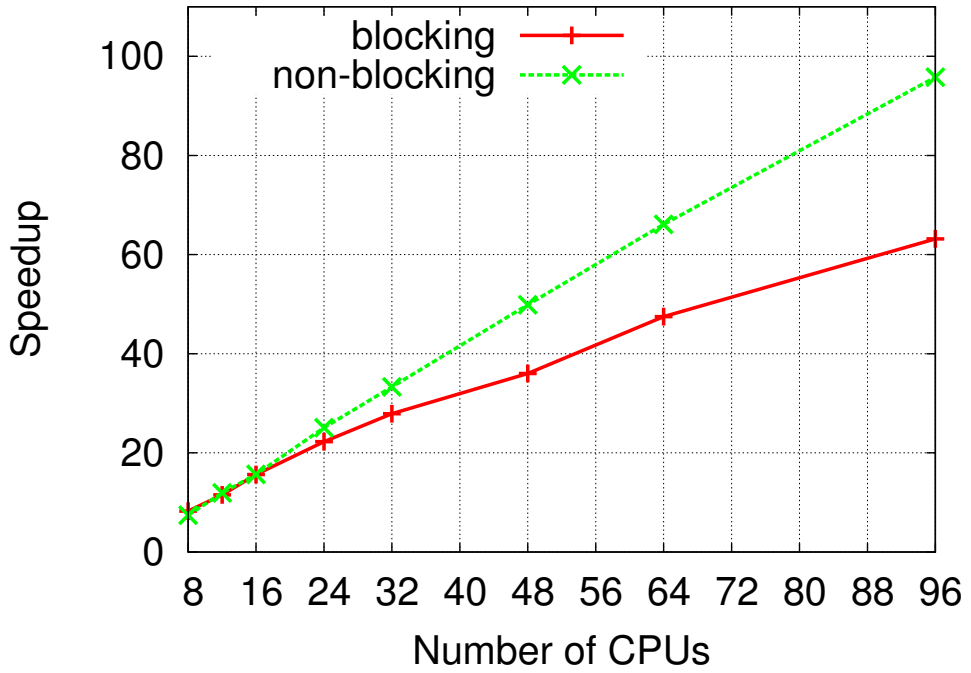
Non-Blocking Collectives for the Matrix-Vector-Product

```
fill_buffers(v_in, send_buffers);
start_send_boundaries(comm_data);
volume_mult(v_in, v_out, comm_data);
finish_send_boundaries(comm_data);
mult_boundaries(v_out, recv_buffers);
```

- `fill_buffers` computes outer elements
- `{start,finish}_send_boundaries` performs overlappable communication
- `volume_mult` is used to overlap communication
- `mult_boundaries` merges the communicated elements

Parallel Speedup (Best Case)

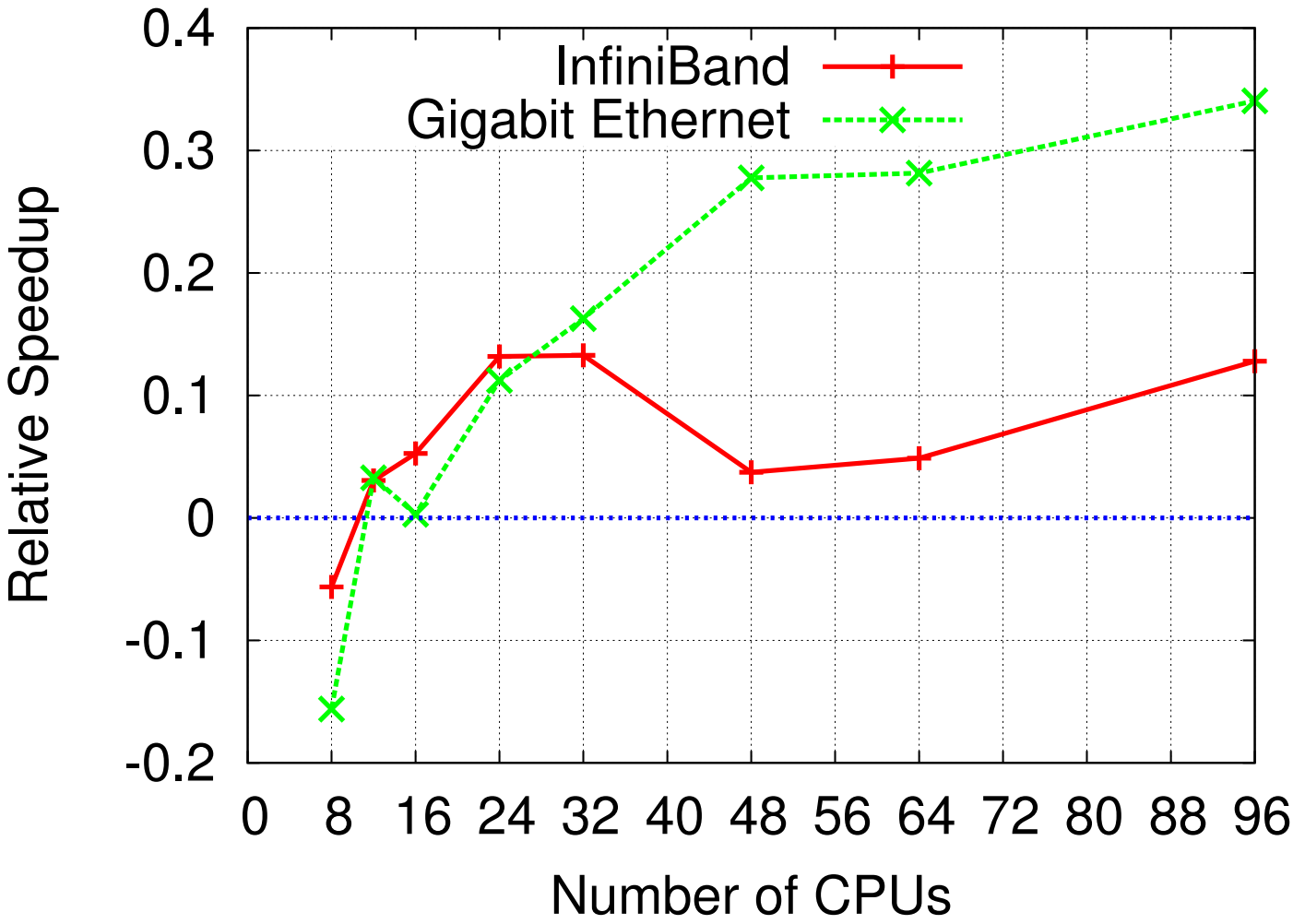




- Cluster: 128 2 GHz Opteron 246 nodes
- Interconnect: Gigabit Ethernet, InfiniBand^{TM}
- System size 800x800x800 (1 node $\approx$ 5300s)

Parallel Gain with Non-Blocking Communication

$$gain = t_{nbl} / t_{bl}$$



Conclusions and Future Work

Conclusions

- overlapping techniques can hide latency
- non-blocking collective operations seem promising
- can be used for parallel CG solvers

Future Work:

- port non-blocking colls into Open MPI
- optimized non-blocking collectives
- more applications and scenarios
- $\Rightarrow$ We would like to collaborate with scientists!

Further Information

`http://www.unixer.de/NBC`

Conclusions and Future Work

Conclusions

- overlapping techniques can hide latency
- non-blocking collective operations seem promising
- can be used for parallel CG solvers

Future Work:

- port non-blocking colls into Open MPI
- optimized non-blocking collectives
- more applications and scenarios
- $\Rightarrow$ We would like to collaborate with scientists!

Further Information

http://www.unixer.de/NBC

Conclusions and Future Work

Conclusions

- overlapping techniques can hide latency
- non-blocking collective operations seem promising
- can be used for parallel CG solvers

Future Work:

- port non-blocking colls into Open MPI
- optimized non-blocking collectives
- more applications and scenarios
- $\Rightarrow$ We would like to collaborate with scientists!

Further Information

`http://www.unixer.de/NBC`