Overlapping Communication and Computation with High Level Communication Routines

- On Optimizing Parallel Applications -

Torsten Hoefler and Andrew Lumsdaine

Open Systems Lab
Indiana University
Bloomington, IN 47405, USA

Conference on Cluster Computing and the Grid (CCGrid'08)
Lyon, France
21th May 2008

Introduction

Solving Grand Challenge Problems

- not a Grid talk
- HPC-centric view
- highly-scalable tightly coupled machines

Thanks for the Introduction Manish!

- All processors will be multi-core
- All computers will be massively parallel
- All programmers will be parallel programmers
- All programs will be parallel programs

$\Rightarrow$ All (massively) parallel programs need optimized communication (patterns)

Fundamental Assumptions (I)

We need more powerful machines!

- Solutions for real-world scientific problems need huge processing power (Grand Challenges)

Capabilities of single PEs have fundamental limits

- The scaling/frequency race is currently stagnating
- Moore's law is still valid (number of transistors/chip)
- Instruction level parallelism is limited (pipelining, VLIW, multi-scalar)

Explicit parallelism seems to be the only solution

- Single chips and transistors get cheaper
- Implicit transistor use (ILP, branch prediction) have their limits

Fundamental Assumptions (II)

Parallelism requires communication

- Local or even global data-dependencies exist
- Off-chip communication becomes necessary
- Bridges a physical distance (many PEs)

Communication latency is limited

- It's widely accepted that the speed of light limits data-transmission
- Example: minimal 0-byte latency for $1m \approx 3.3ns \approx 13$ cycles on a $4GHz$ PE

Bandwidth can hide latency only partially

- Bandwidth is limited (physical constraints)
- The problem of "scaling out" (especially iterative solvers)

Assumptions about Parallel Program Optimization

Collective Operations

- Collective Operations (COs) are an optimization tool
- CO performance influences application performance
- optimized implementation and analysis of CO is non-trivial

Hardware Parallelism

- More PEs handle more tasks in parallel
- Transistors/PEs take over communication processing
- Communication and computation could run simultaneously

Overlap of Communication and Computation

- Overlap can hide latency
- Improves application performance

Overview (I)

Theoretical Considerations

- a model for parallel architectures
- parametrize model
- derive model for BC and NBC
- prove optimality of collops in the model (?)
- show processor idle time during BC
- show limits of the model (IB,BG/L)

Implementation of NBC

- how to assess performance?
- highly portable low-performance
- IB optimized, high performance, threaded

Overview (II)

Application Kernels

- FFT (strong data dependency)
- compression (parallel data analysis)
- poisson solver (2d-decomposition)

Applications

- show how performance benefits for microbenchmarks can benefit real-world applications
- ABINIT
- Octopus
- OSEM medical image reconstruction

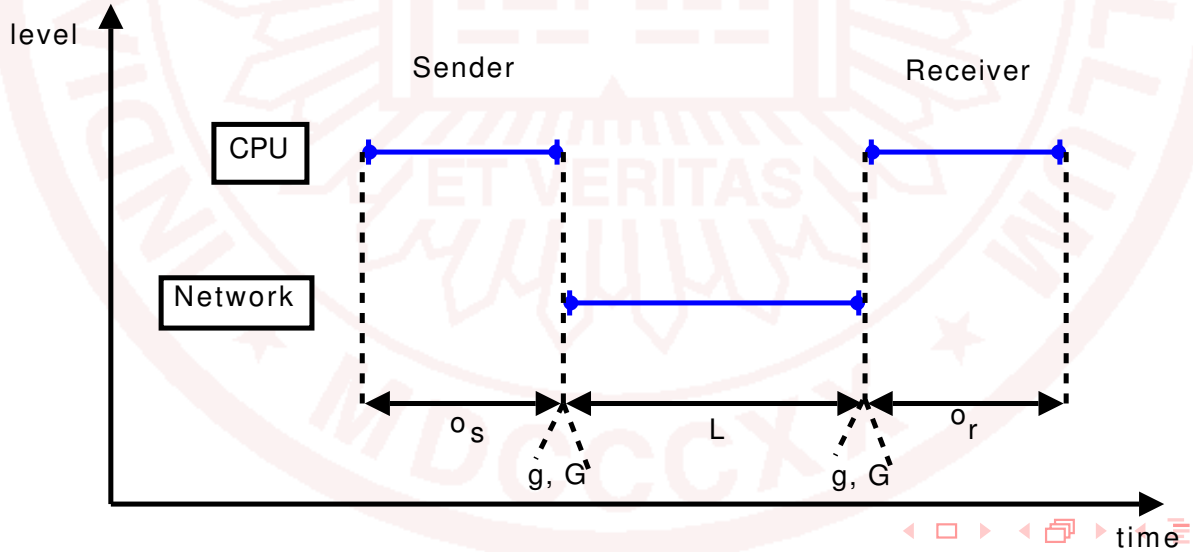
The LogGP model

Modelling Network Communication

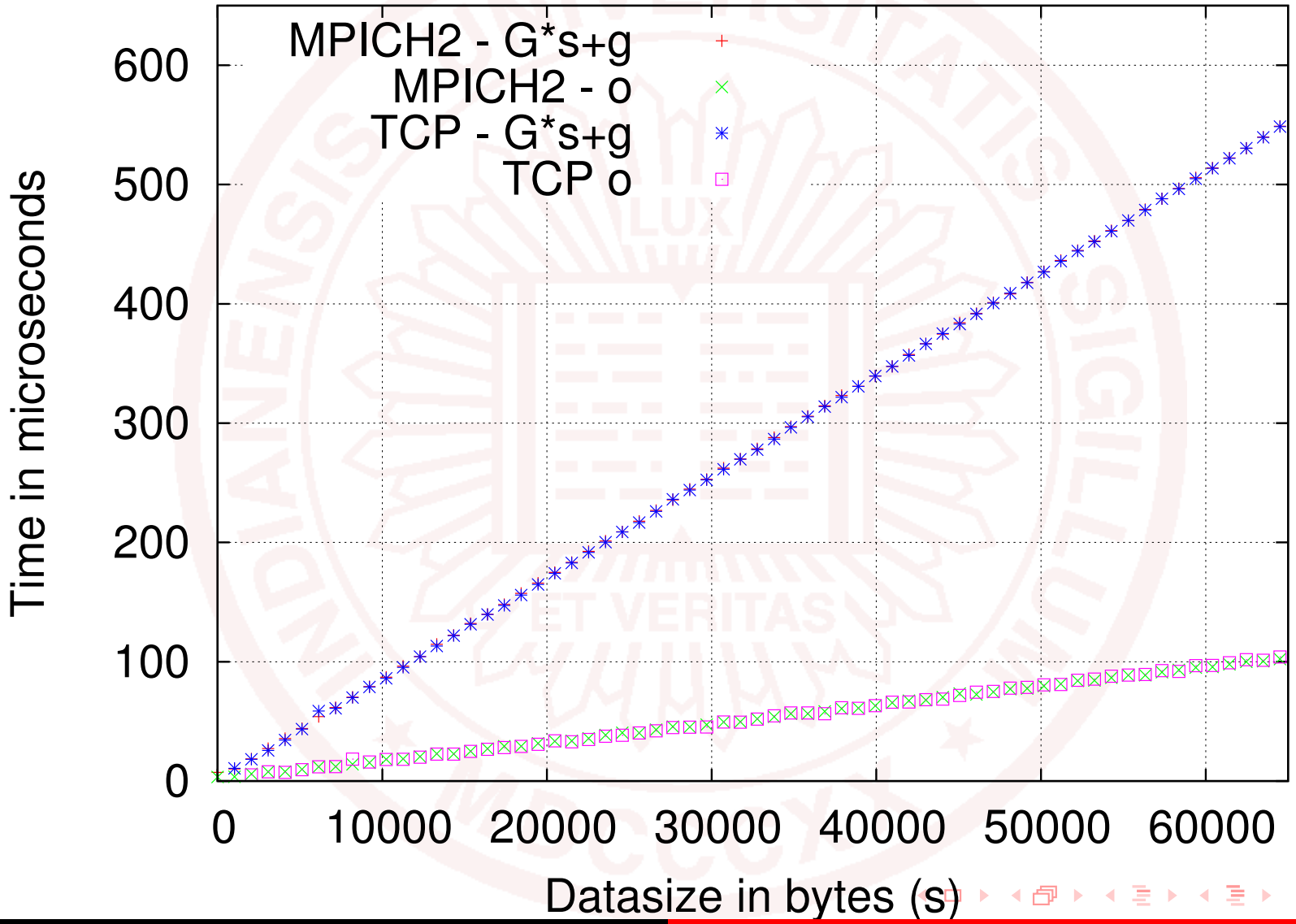
- LogP model family has best tradeoff between ease of use and accuracy
- LogGP is most accurate for different message sizes

Methodology

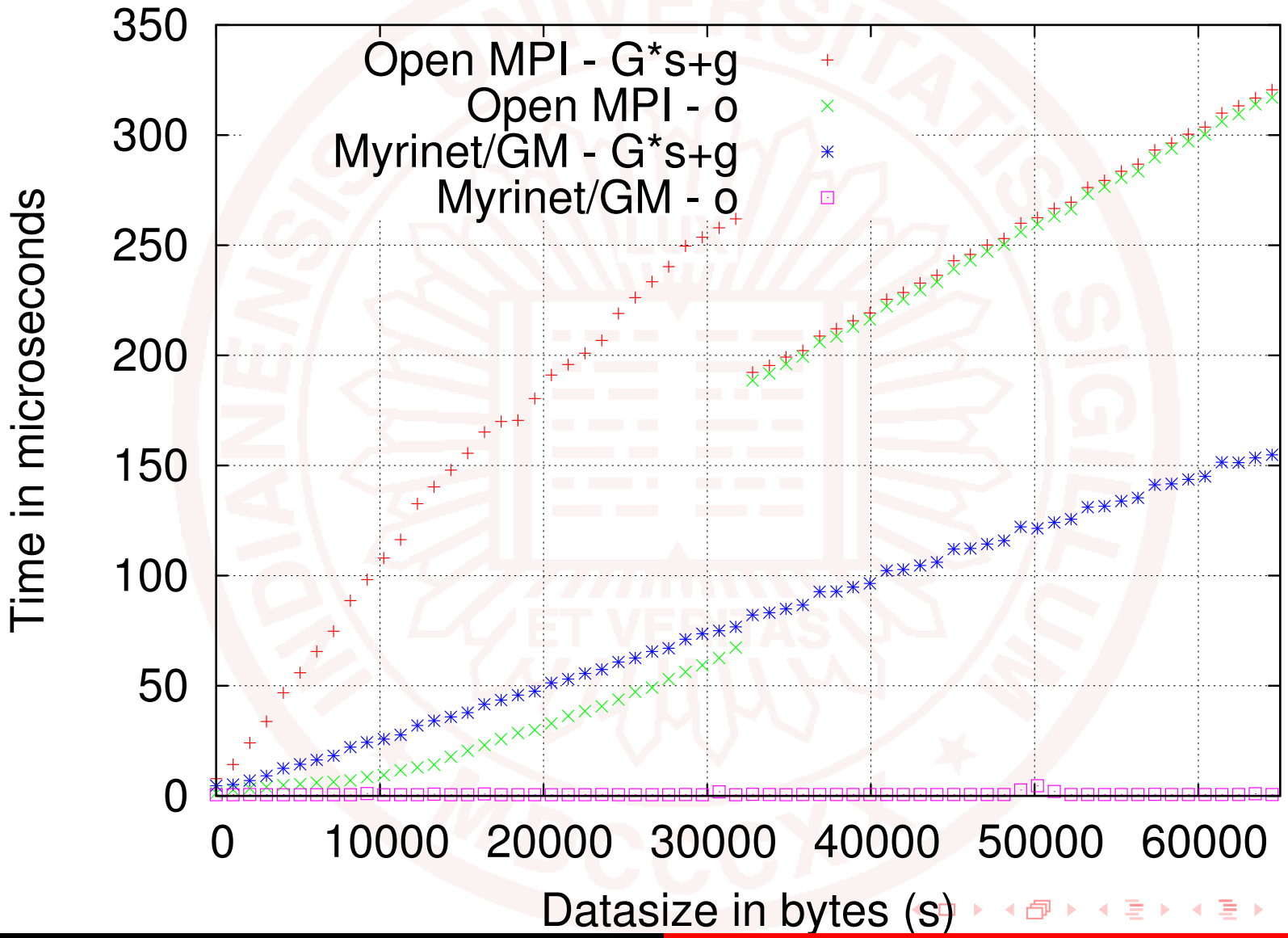
- assess LogGP parameters for modern interconnects
- model collective communication



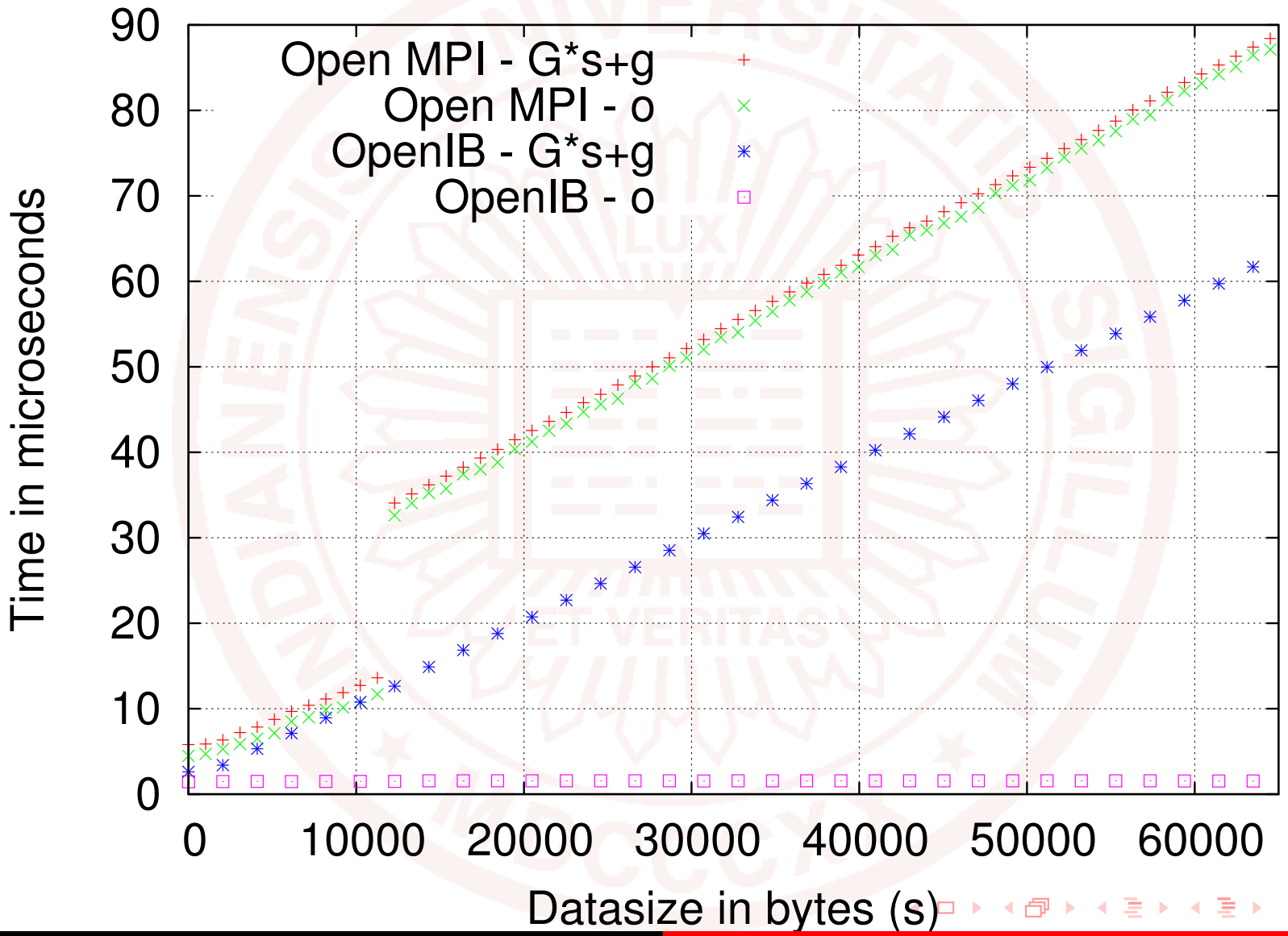
TCP/IP - GigE/SMP



Myrinet/GM (preregistered/cached)



InfiniBand (preregistered/cached)



Modelling Collectives

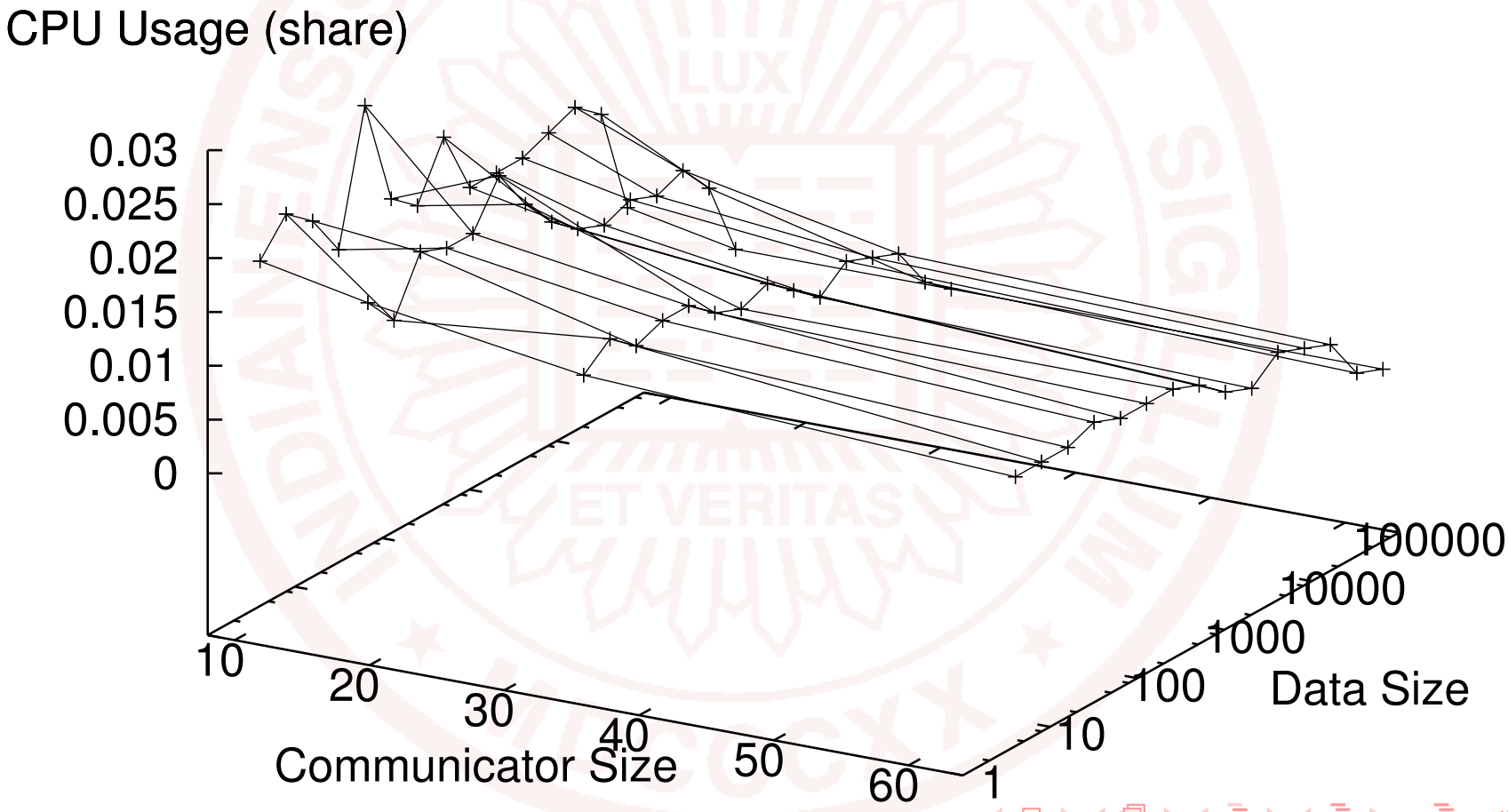
LogGP Models - general

$$
\begin{aligned}
t_{barr} &= (2o + L) \cdot \lceil log_2 P \rceil \\
t_{allred} &= 2 \cdot (2o + L + m \cdot G) \cdot \lceil log_2 P \rceil + m \cdot \gamma \cdot \lceil log_2 P \rceil \\
t_{bcast} &= (2o + L + m \cdot G) \cdot \lceil log_2 P \rceil
\end{aligned}
$$

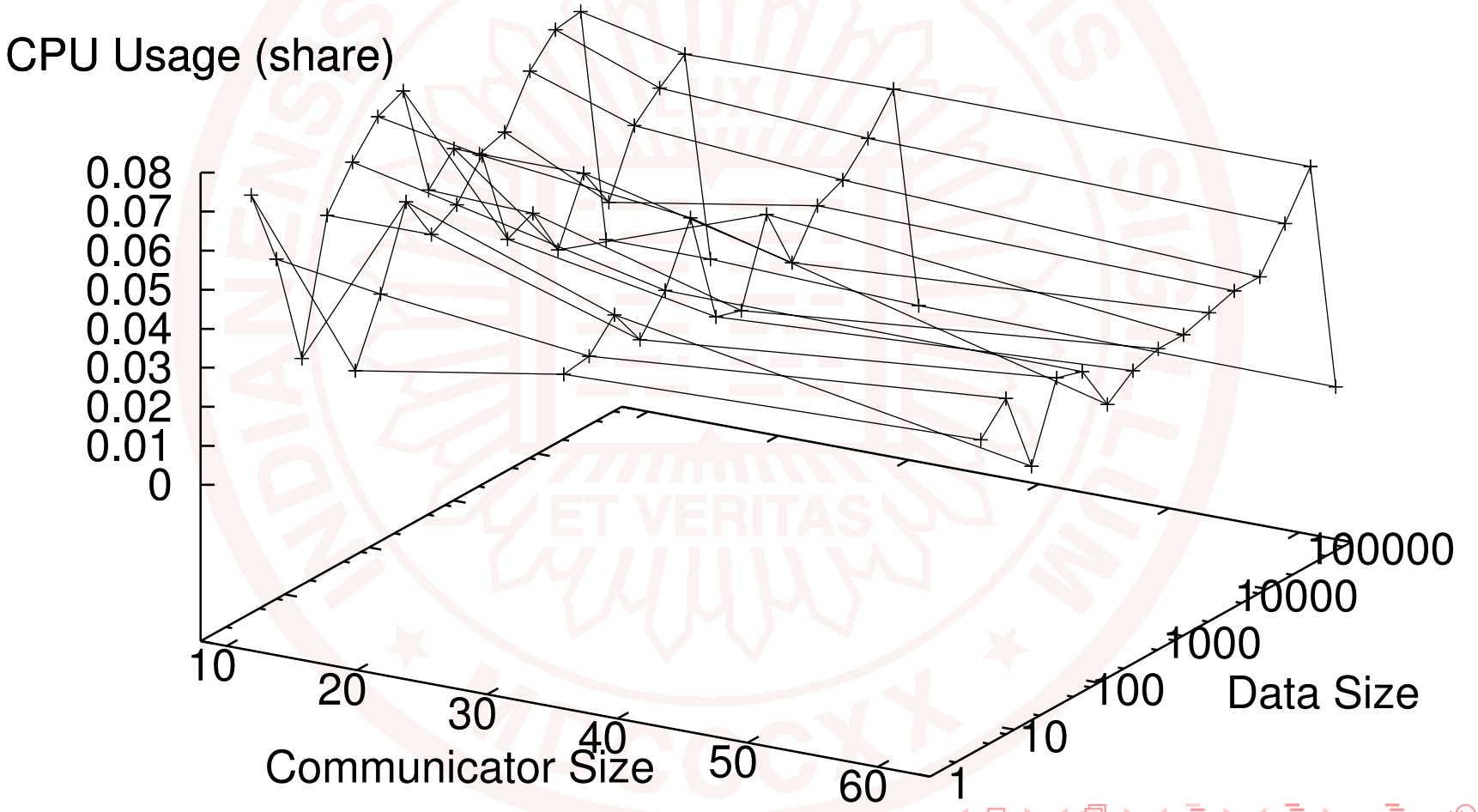
CPU and Network LogGP parts

$$
\begin{aligned}
t_{barr}^{CPU} &= 2o \cdot \lceil log_2 P \rceil & t_{barr}^{NET} &= L \cdot \lceil log_2 P \rceil \\
t_{allred}^{CPU} &= (4o + m \cdot \gamma) \cdot \lceil log_2 P \rceil & t_{allred}^{NET} &= 2 \cdot (L + m \cdot G) \cdot \lceil log_2 P \rceil \\
t_{bcast}^{CPU} &= 2o \cdot \lceil log_2 P \rceil & t_{bcast}^{NET} &= (L + m \cdot G) \cdot \lceil log_2 P \rceil
\end{aligned}
$$

CPU Overhead - MPI_Allreduce LAM/MPI 7.1.2



CPU Overhead - MPI_Allreduce MPICH2 1.0.3



Implementation of Non-blocking Collectives

LibNBC for MPI

- single-threaded
- highly portable
- schedule-based design

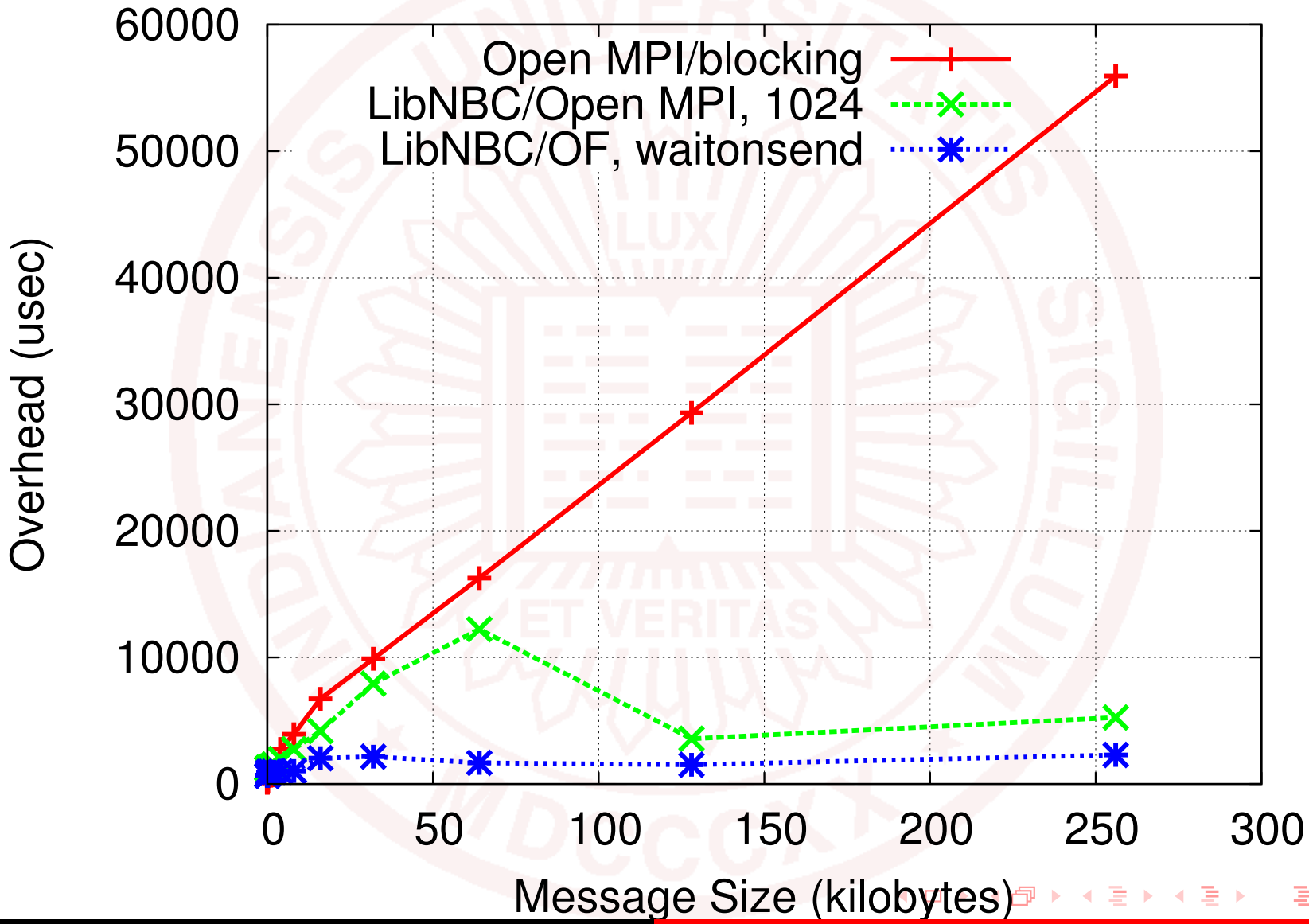
LibNBC for InfiniBand

- single-threaded (first version)
- receiver-driven message passing
- very low overhead

Threaded LibNBC

- thread support requires MPI_THREAD_MULTIPLE
- completely asynchronous progress
- complicated due to scheduling issues

LibNBC - Alltoall overhead, 64 nodes



First Example

Derivation from “normal” implementation

- distribution identical to “normal” 3D-FFT
- first FFT in z direction and index-swap identical

Design Goals to Minimize Communication Overhead

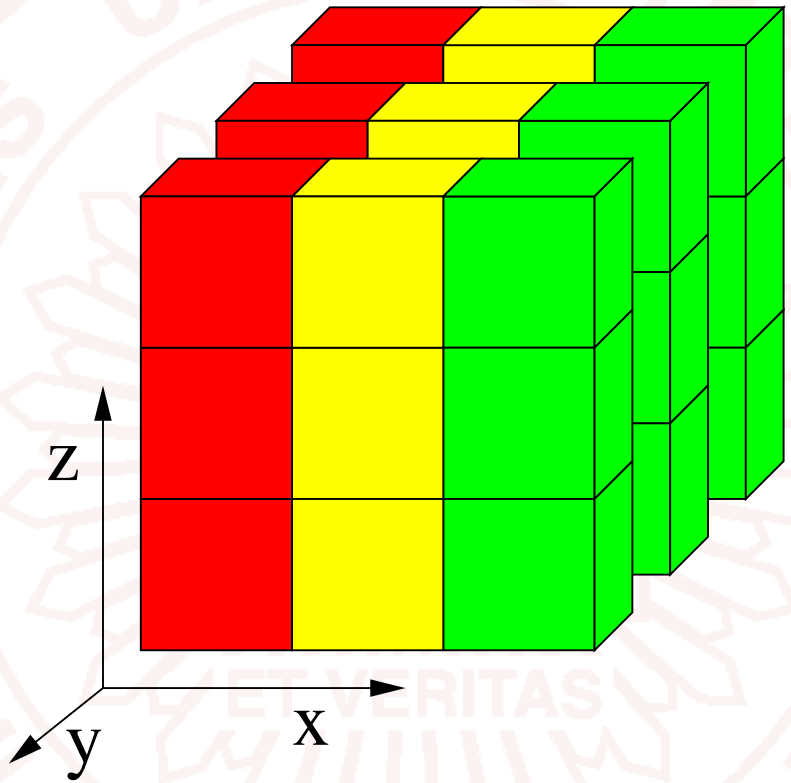
- start communication as early as possible
- achieve maximum overlap time

Solution

- start MPI_Ialltoall as soon as first xz-plane is ready
- calculate next xz-plane
- start next communication accordingly ...
- collect multiple xz-planes (tile factor)

Transformation in z Direction

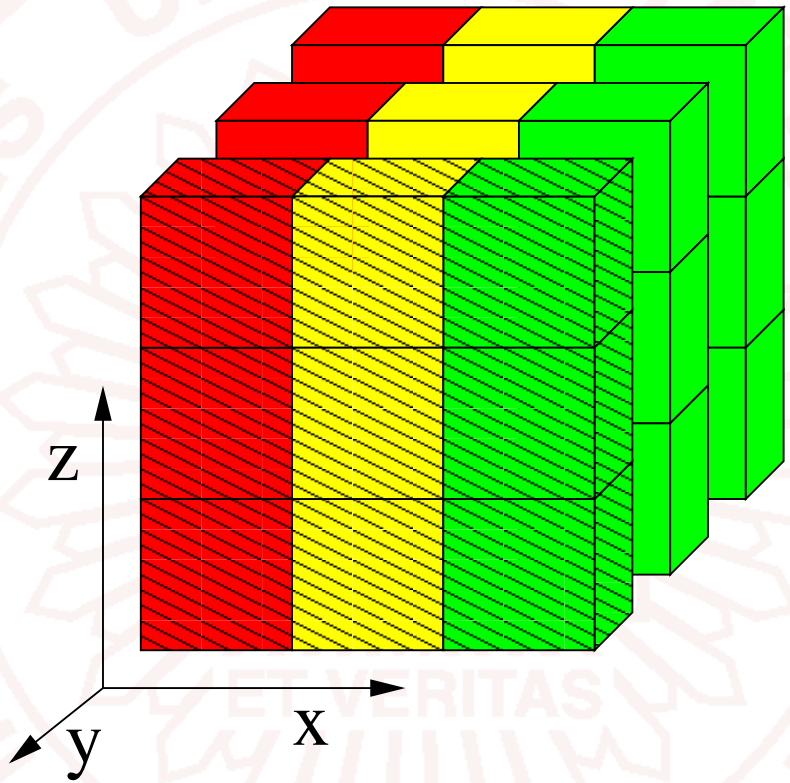
Data already transformed in y direction



1 block = 1 double value (3x3x3 grid)

Transformation in z Direction

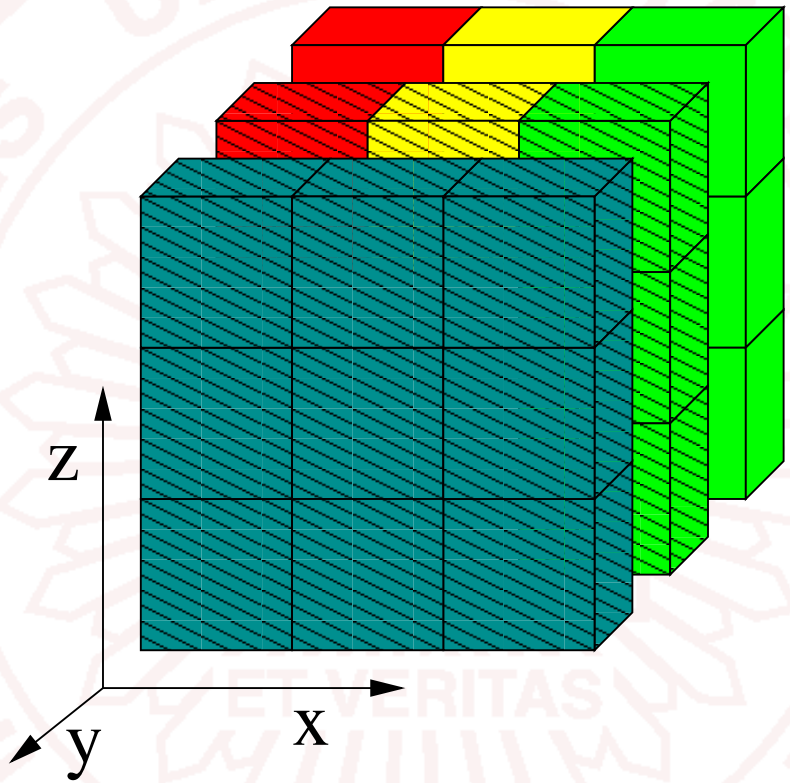
Transform first xz plane in z direction



pattern means that data was transformed in y and z direction

Transformation in z Direction

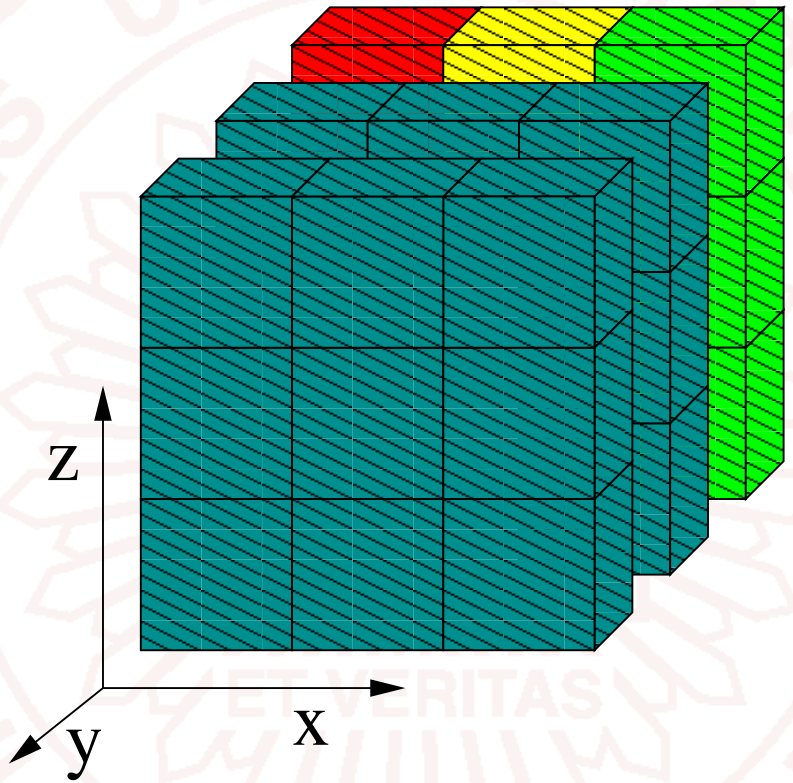
start MPI_Ialltoall of first xz plane and transform second plane



cyan color means that data is communicated in the background

Transformation in z Direction

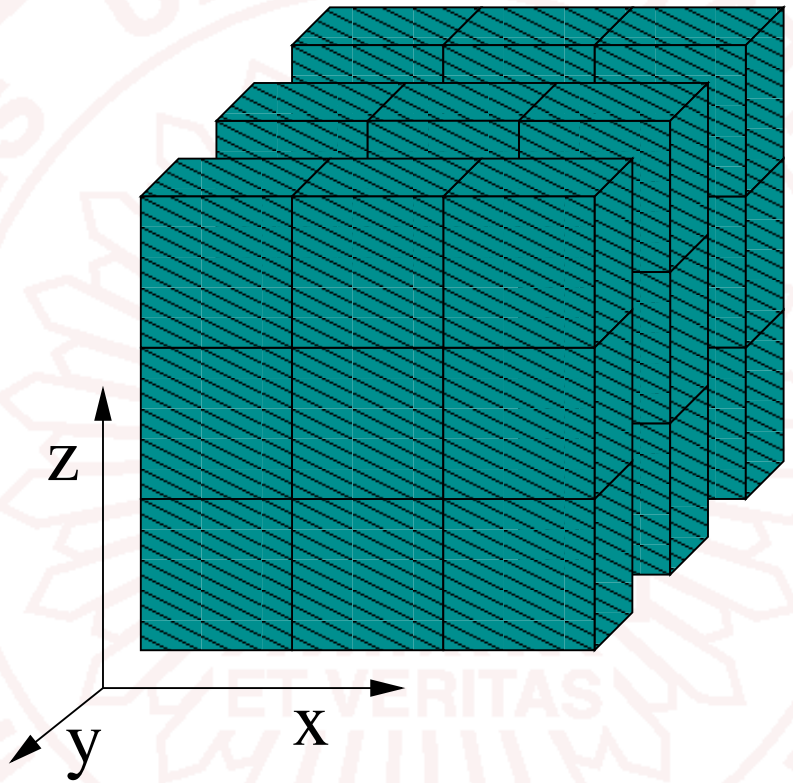
start MPI_Ialltoall of second xz plane and transform third plane



data of two planes is not accessible due to communication

Transformation in x Direction

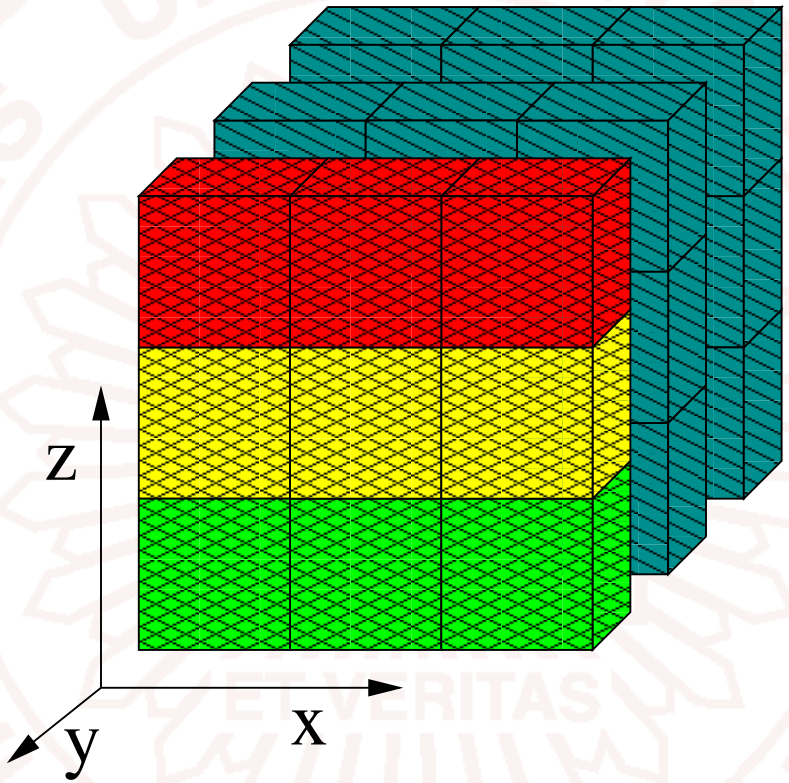
start communication of the third plane and ...



we need the first xz plane to go on ...

Transformation in x Direction

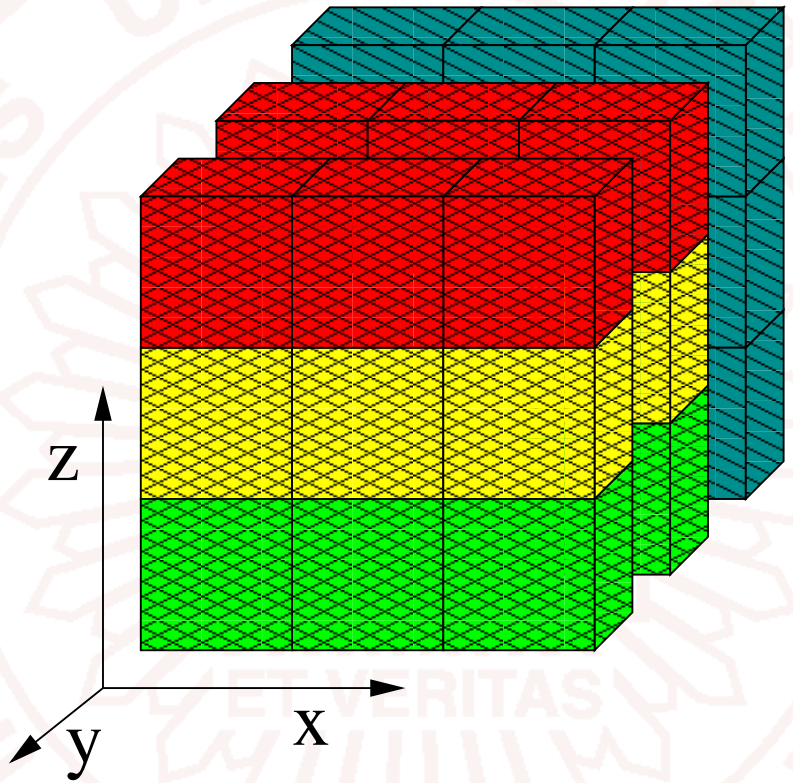
... so MPI_Wait for the first MPI_Ialltoall!



and transform first plane (new pattern means xyz transformed)

Transformation in x Direction

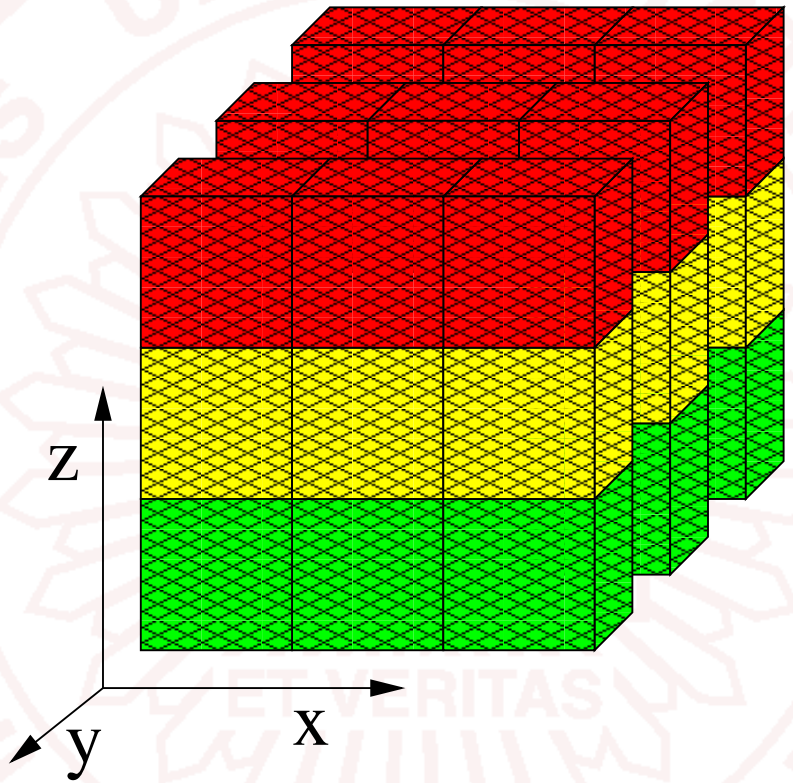
Wait and transform second xz plane



first plane's data could be accessed for next operation

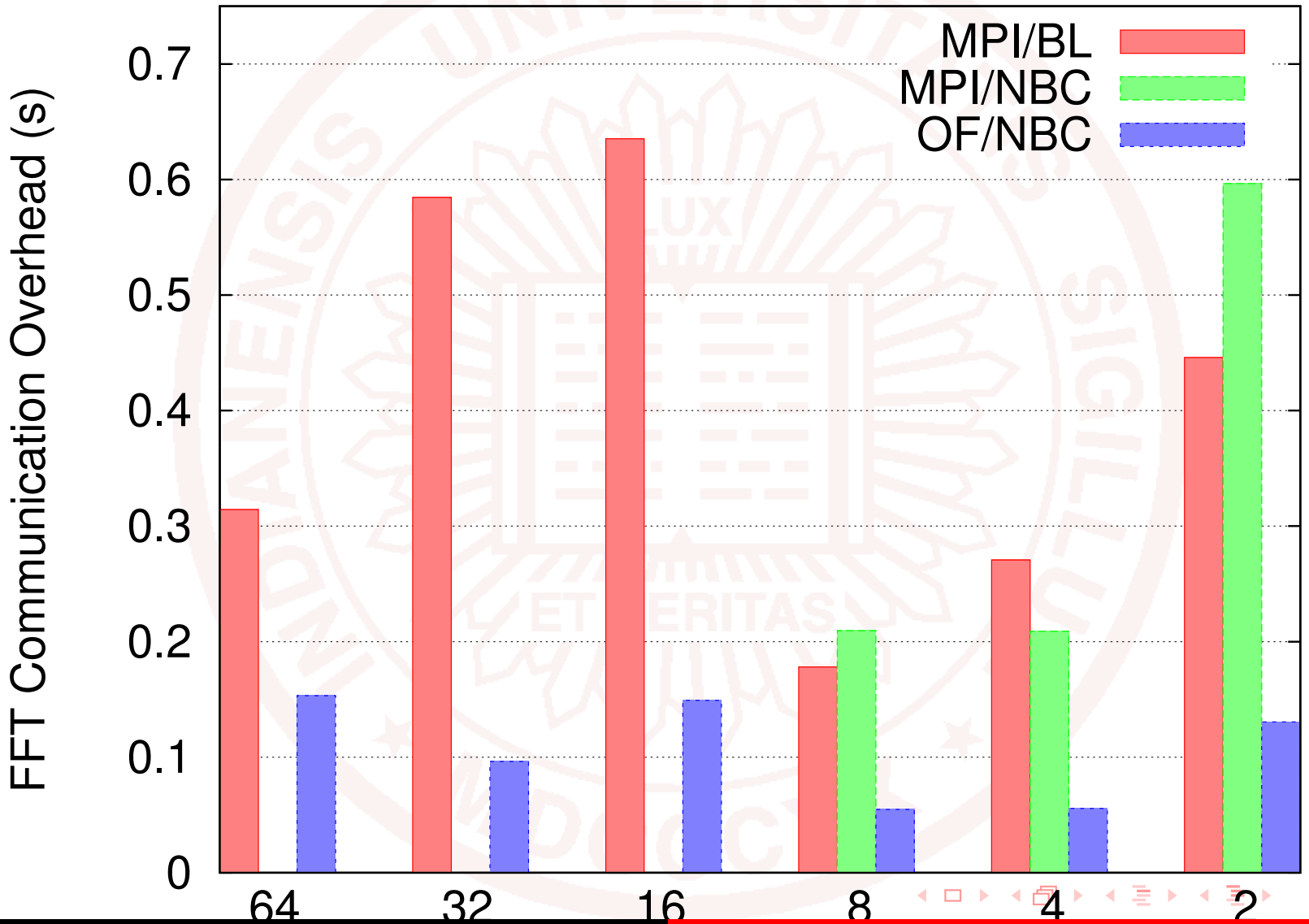
Transformation in x Direction

wait and transform last xz plane



done! $\rightarrow$ 1 complete 1D-FFT overlaps a communication

3d-FFT performance

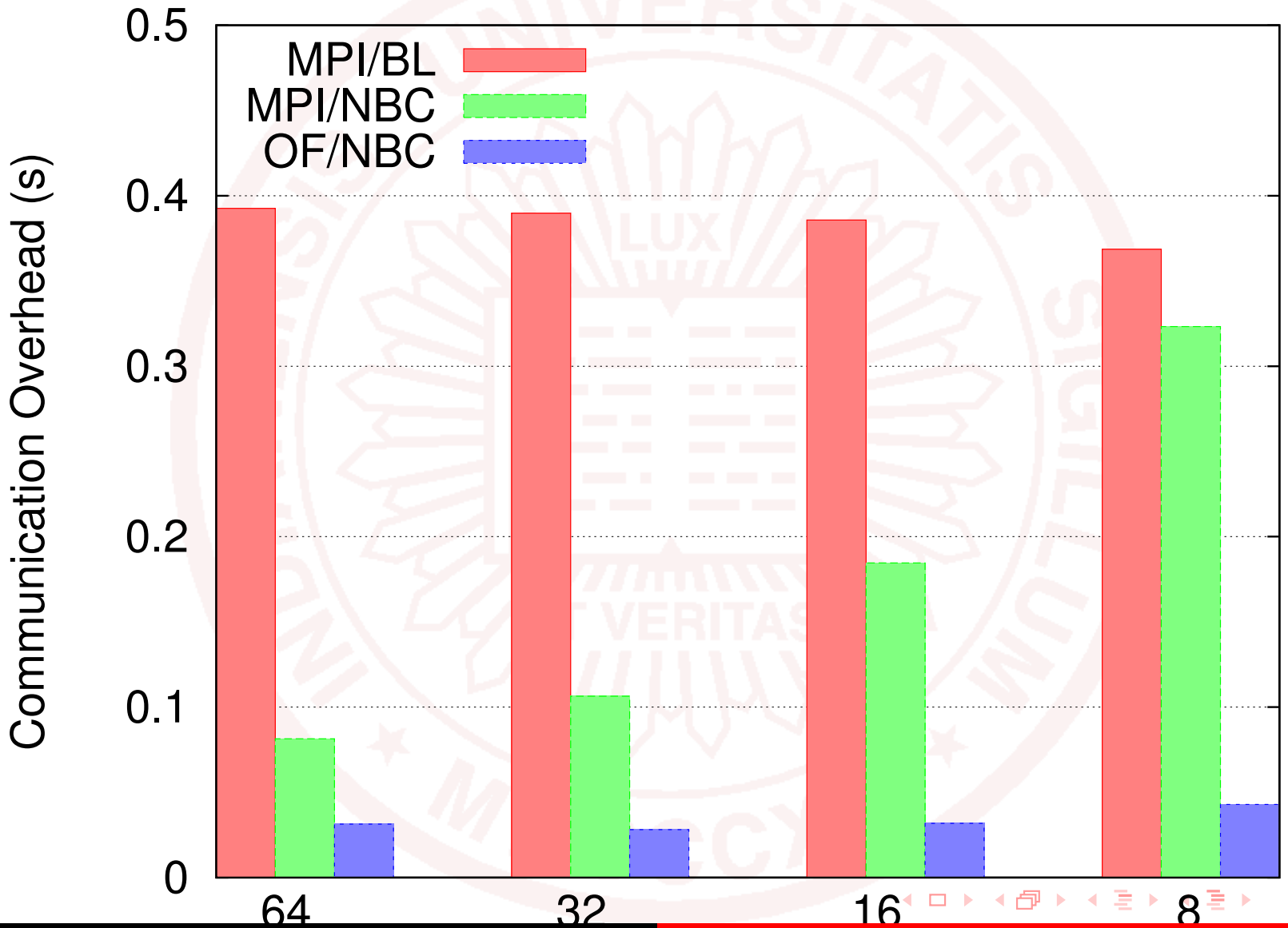


Second Example

Data Parallel Loops - Parallel Compression

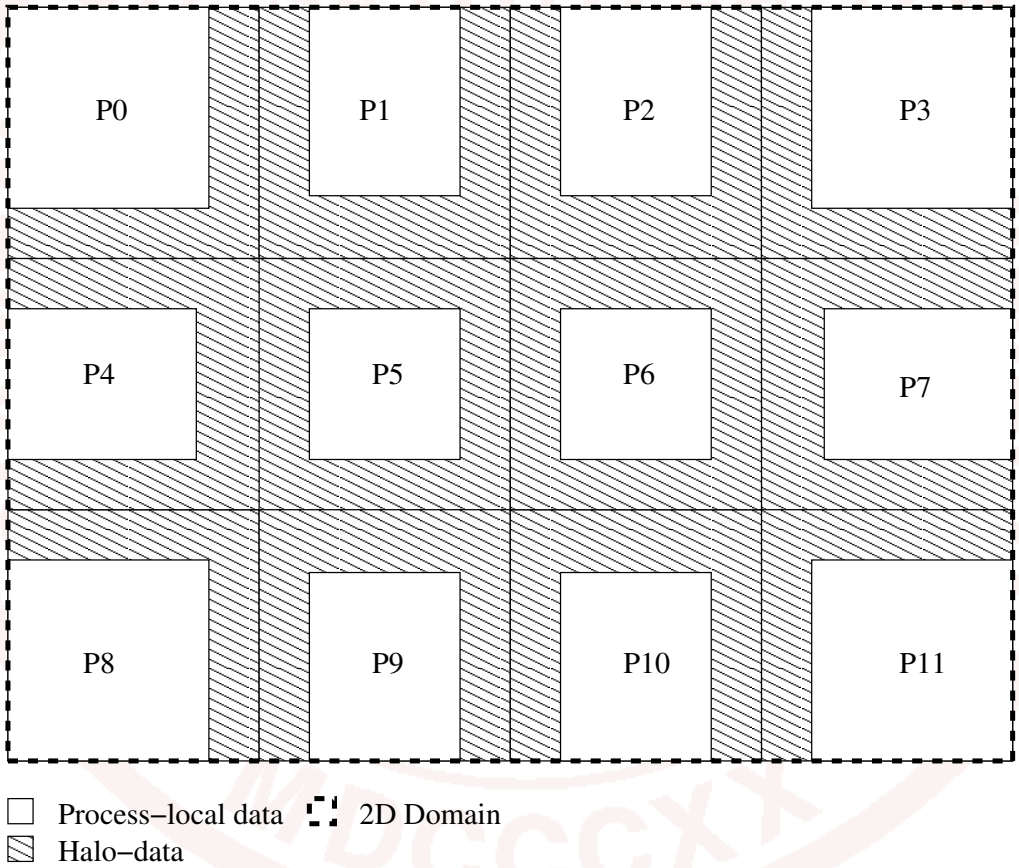
```
for (i=0; i < N/P; i++) {
  compute(i);
}
comm(N/P);
```

Parallel Compression Performance

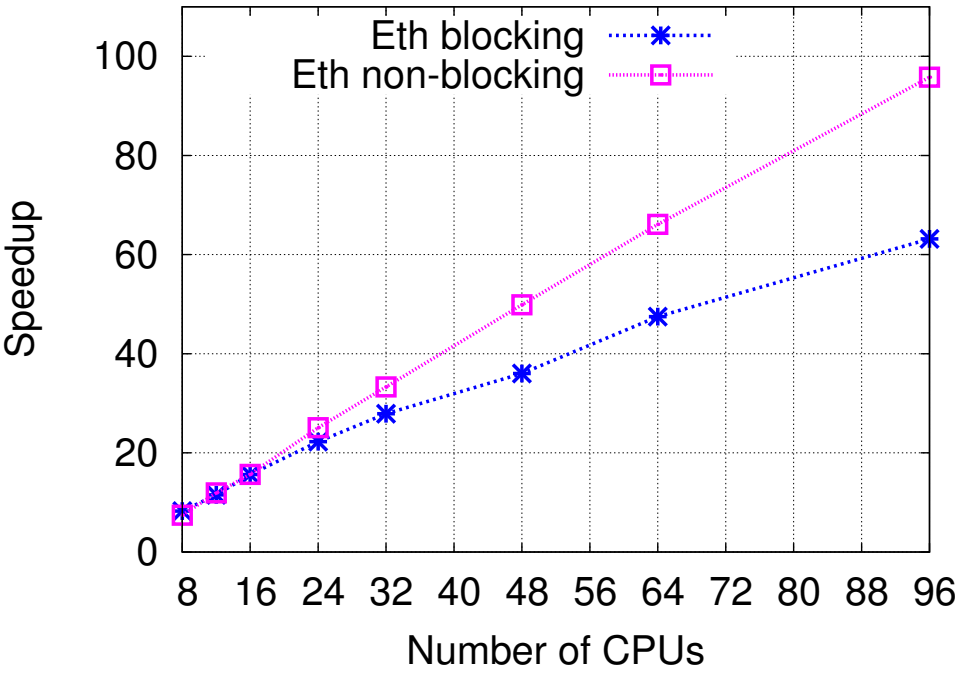


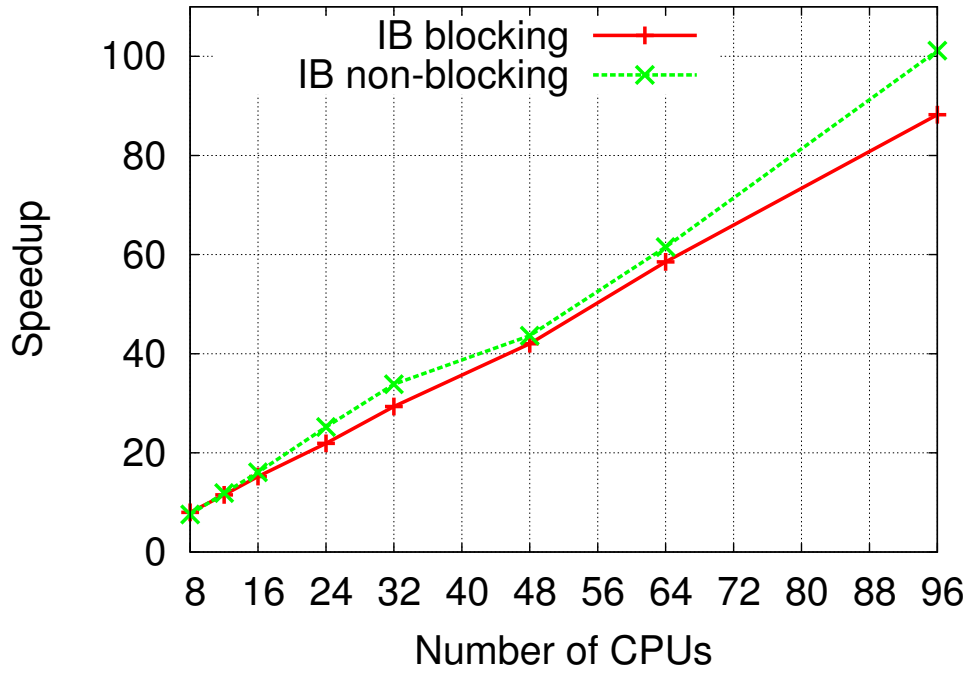
Domain Decomposition

- nearest neighbor communication
- can be implemented with MPI_Alltoallv
- we propose new collective MPI_Neighbor_xchg[v]



Parallel 3d-Poisson solver - Speedup

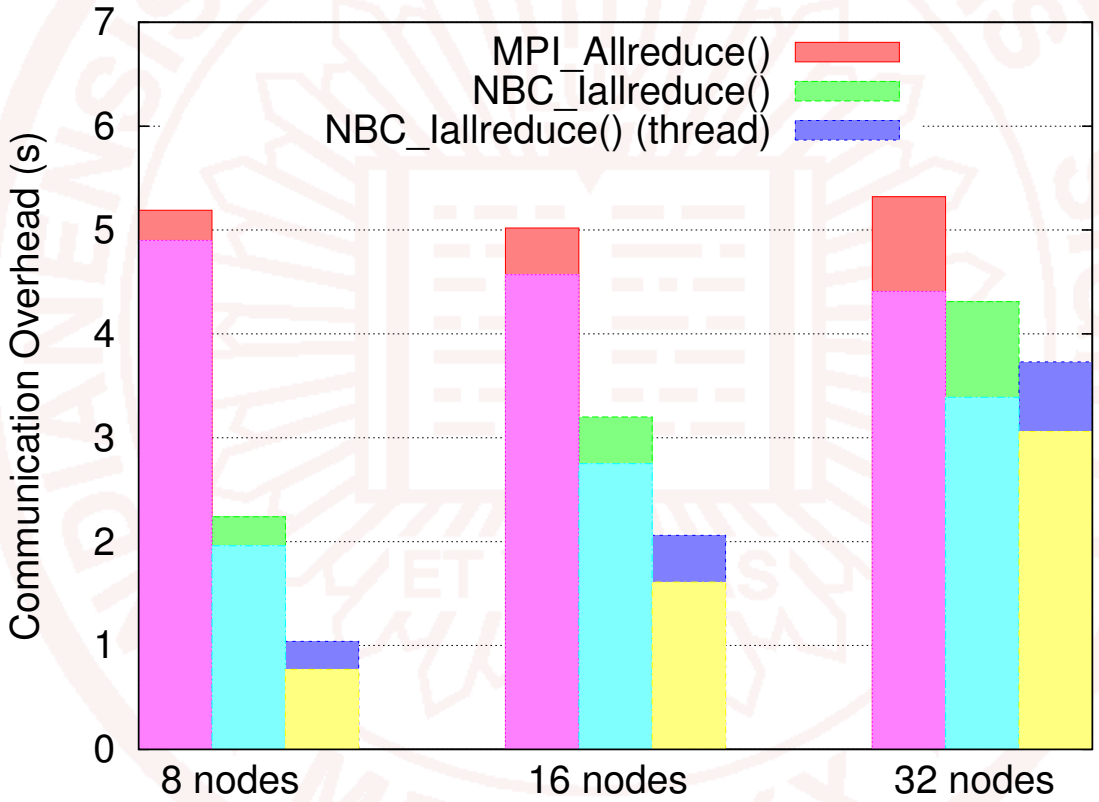




- Cluster: 128 2 GHz Opteron 246 nodes
- Interconnect: Gigabit Ethernet, InfiniBand™
- System size 800x800x800 (1 node ≈ 5300s)

Medical Image Reconstruction

- OSEM algorithm
- Allreduction of full image



Thank you for your Attention

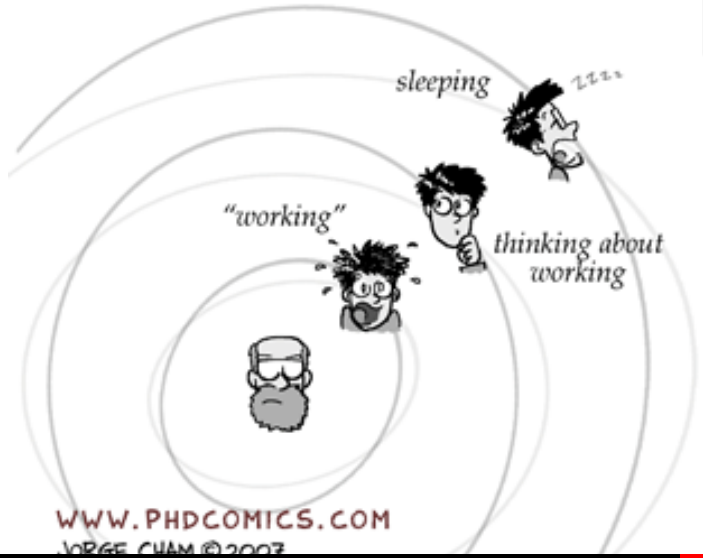
An Introduction to

QUANTUM Gradnamics

"I discard all hope of predicting hitherto unpredictable quantities, such as my graduation."
- Werner Heisenberg

Another principal concept in Quantum Gradnamics is the observation that graduate students do not move toward graduation in a steady and continuous manner. Rather, they make progress through discrete bursts of random productivity called "wanta" (short for "want data") whose energy is proportional to the frequency of meetings with their advisor.

Grad students, or "p-ons" as Einstein called them, can only occupy a discrete number of energy states:



A direct consequence of this is the "Heisenberg Uncertainty Principle", perhaps the most well-known theorem of Quantum Gradnamics. Developed by Heisenberg during a particularly unproductive period in his graduate career, the principle states that it is not possible to know where a grad student is and where it is going at the same time:

no idea what they are doing × *no idea what they're doing with the rest of their lives* > *Normal amount of Uncertainty*

When probed under pressure, a grad student will either blurt out what they are doing (but won't know if it means anything), or they will blurt out what they *plan* to do (but won't know *how* to do it). Simply put, there is an inherent degree of certainty and precision that is missing from their everyday life.

Heisenberg attributed this to the fact that meetings with professors are *non-communicative* (that is, the order in which orders are given doesn't tell you whether they are worth doing).