Improving the Performance of the Scaled Matrix/Vector Multiplication with Vector Addition in Tau3P, an Electromagnetic Solver

September 8, 2004

Michael M. Wolf, University of Illinois, Urbana-Champaign; Ali Pinar and Esmond G. Ng, Lawrence Berkeley National Laboratory

Outline

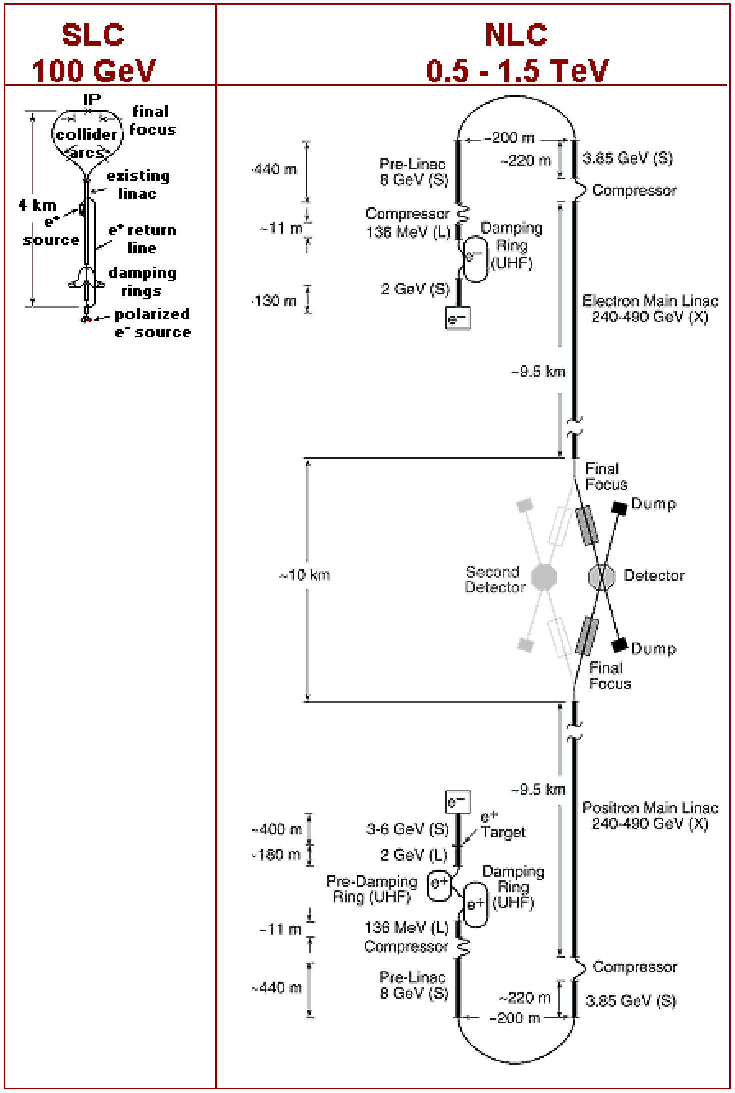
- Motivation
- Brief Description of Tau3P
- Tau3P Performance
- MPI 2-Sided Communication
- Basic Algorithm
- Implementation
- Results
- Conclusions
- Future Work

Challenges in E&M Modeling of Accelerators

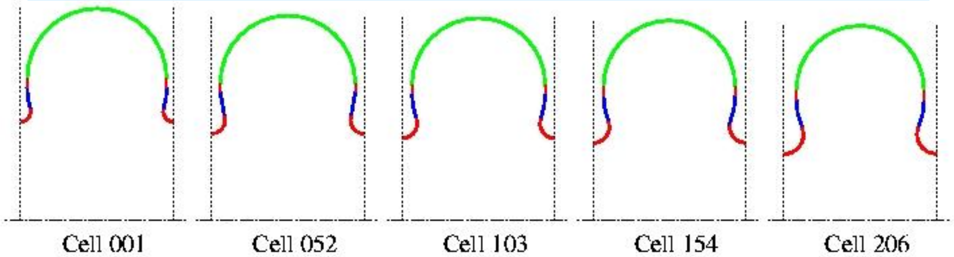
- Accurate modeling essential for modern accelerator design
  - Reduces Design Cost
  - Reduces Design Cycle
- Conformal meshes (Unstructured grid)
- Large, complex electromagnetic structures
  - 100's of millions of DOFs
- Small beam size
  - Large number of mesh points
  - Long run time
- Parallel Computing needed (time and storage)

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Next Linear Collider (NLC)

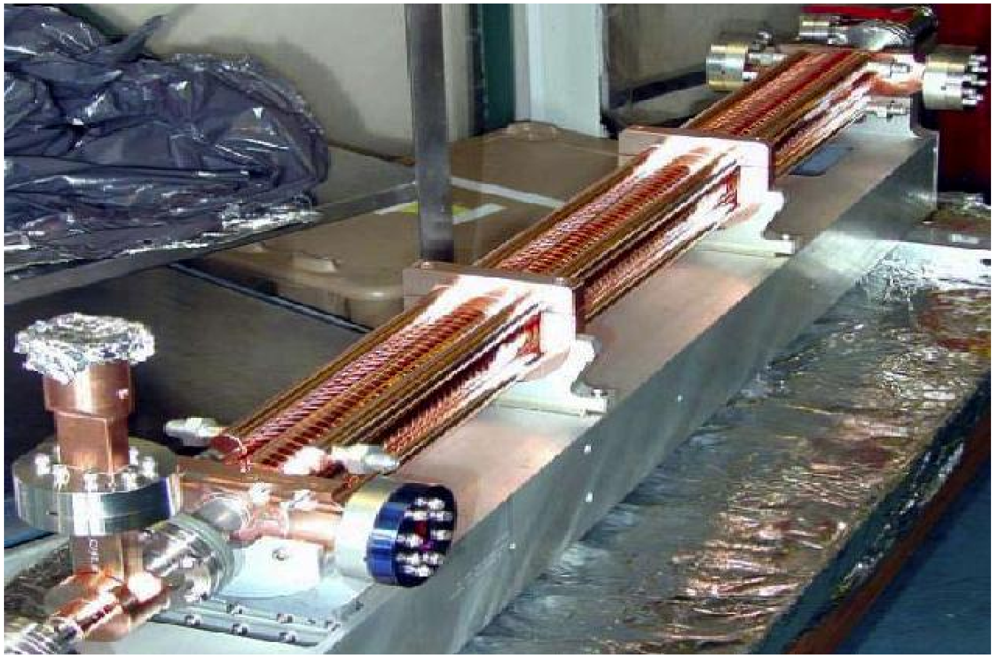


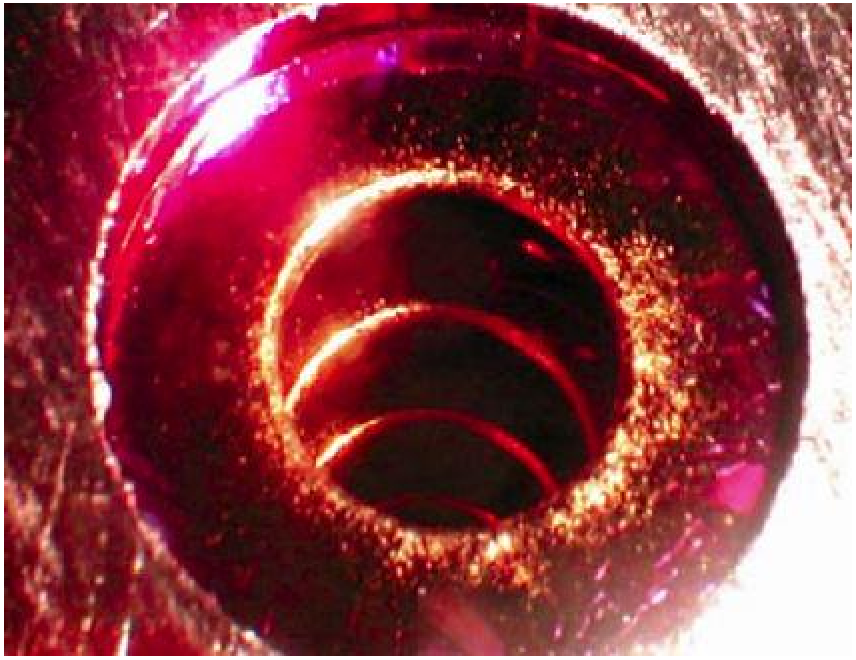
Cell to cell variation of order microns to suppress short range wakes by detuning



End-to-end NLC Structure Simulation

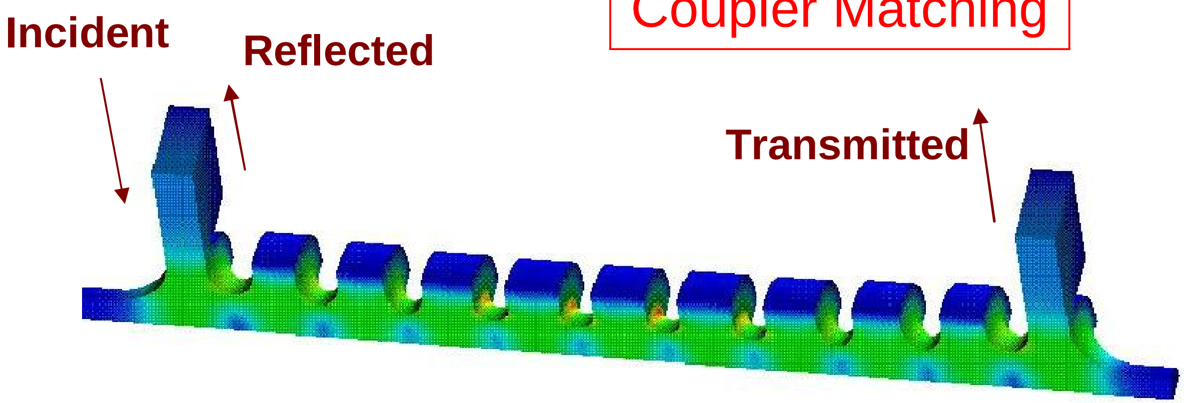
- NLC X-band structure showing damage in the structure cells after high power test
- Theoretical understanding of underlying processes lacking so realistic simulation is needed

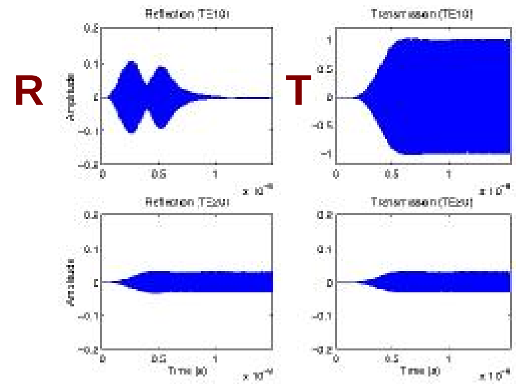




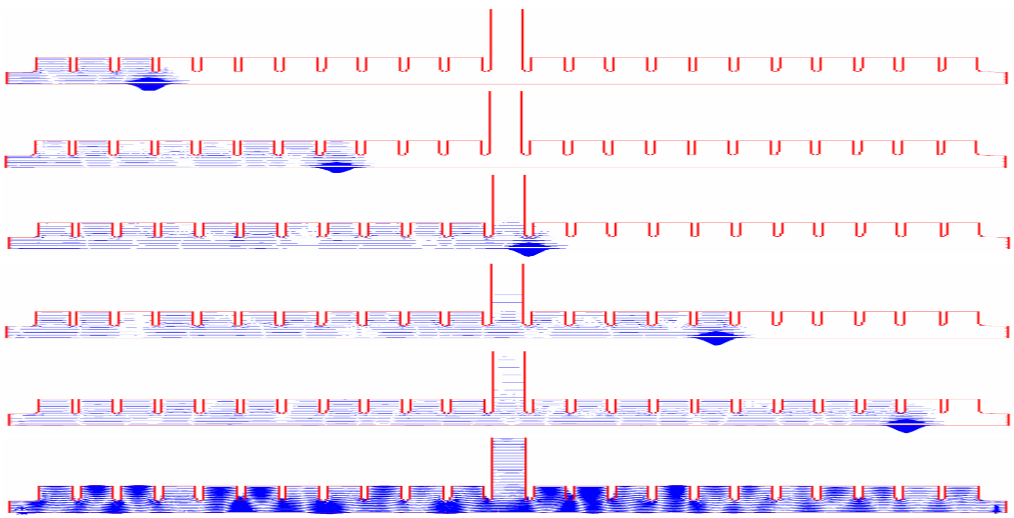
Parallel Time-Domain Field Solver – Tau3P

Coupler Matching

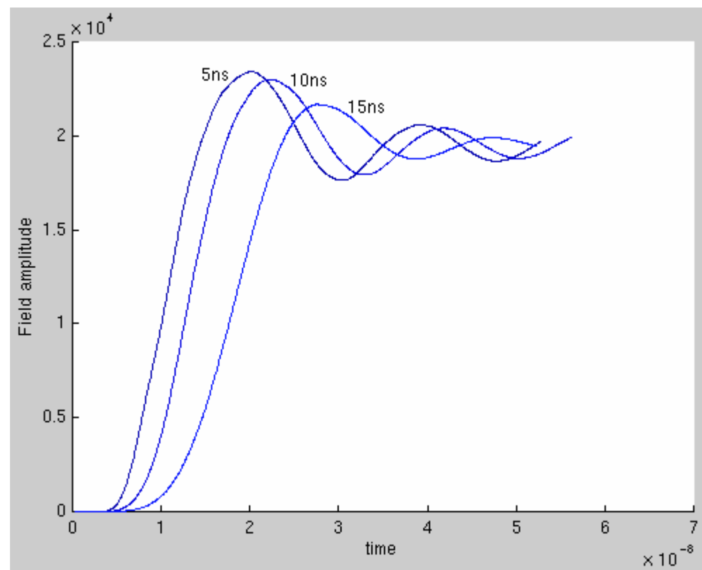




Wakefield Calculations



Rise Time Effects

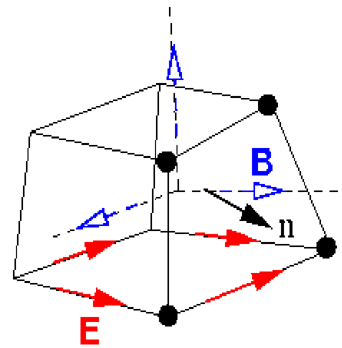


Parallel Time-Domain Field Solver – Tau3P

- Follows evolution of E and H fields inside accelerator cavity
- DSI method on non-orthogonal meshes

$$\oint E \bullet ds = -\iint \frac{\partial B}{\partial t} \bullet dA$$

$$\oint H \bullet ds^* = \iint \frac{\partial D}{\partial t} \bullet dA^* + \iint j \bullet dA^*$$



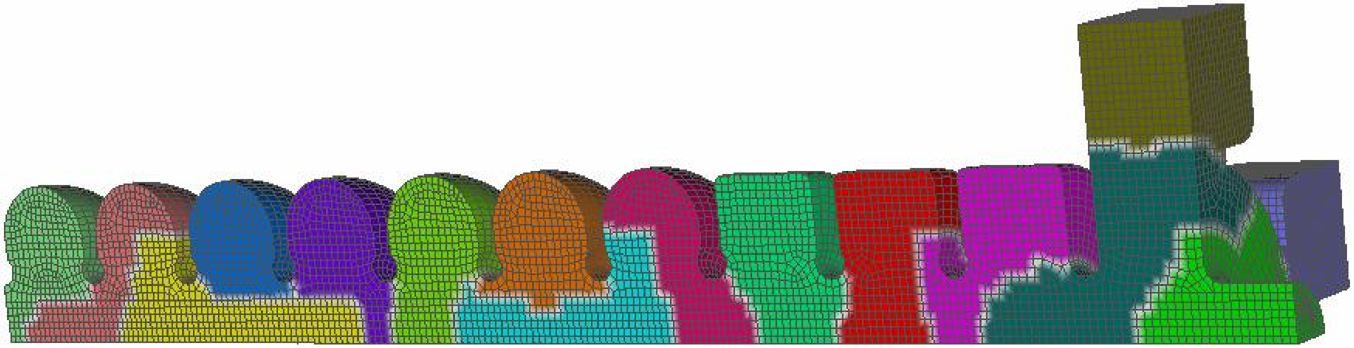
The DSI formulation yields:

$$\vec{e} += a \cdot A_H \cdot \vec{h}$$

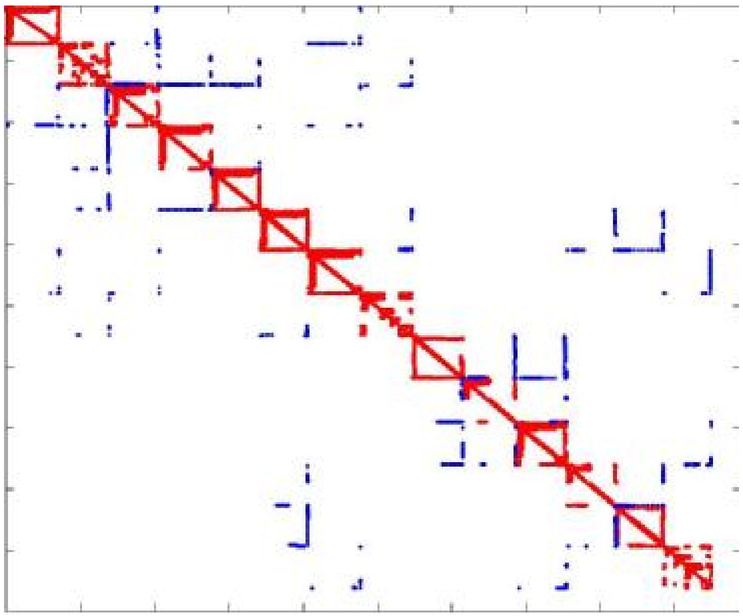
$$\vec{h} += b \cdot A_E \cdot \vec{e}$$

- α, β are constants proportional to dt
- A_H, A_E are matrices
- Electric fields on primary grid
- Magnetic fields on embedded dual grid
- Leapfrog time advancement
- (FDTD) for orthogonal grids

Tau3P Implementation



Example of Distributed Mesh



Typical Distributed Matrix

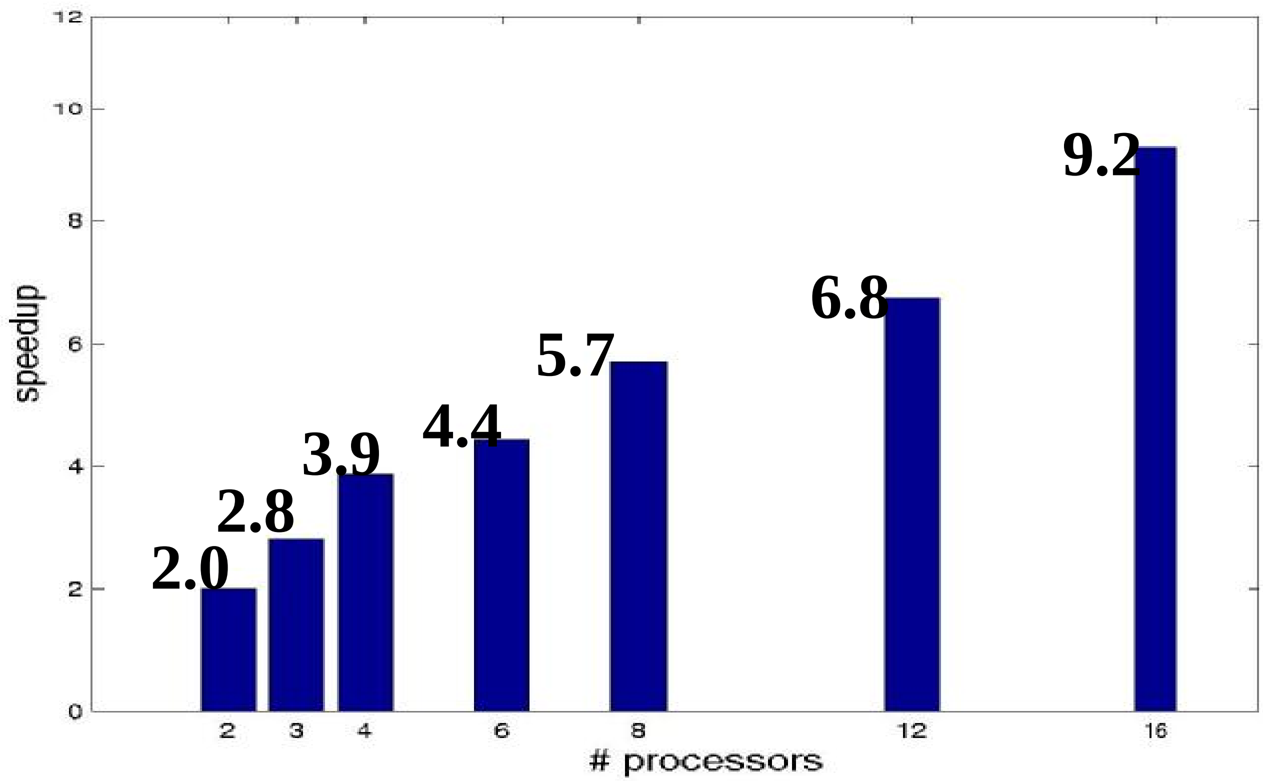
- Very Sparse Matrices
  - 4-20 nonzeros per row
- 2 Coupled Matrices (A_H,A_E)
- Nonsymmetric (Rectangular)

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Parallel Performance of Tau3P (ParMETIS)

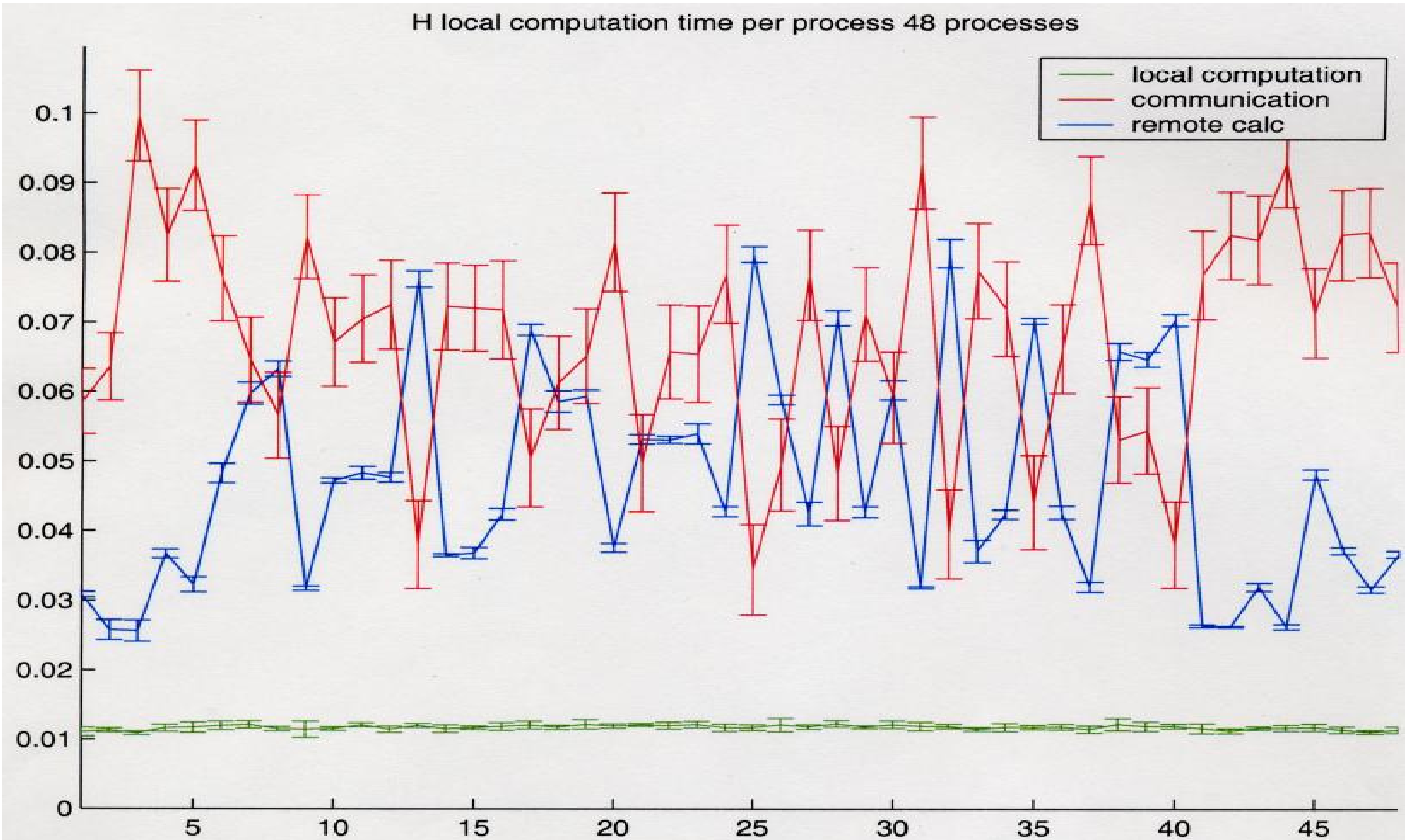
- 257K hexahedrons
- 11.4 million non-zeroes

Parallel Speedup



Communication in Tau3P (ParMETIS Partitioning)

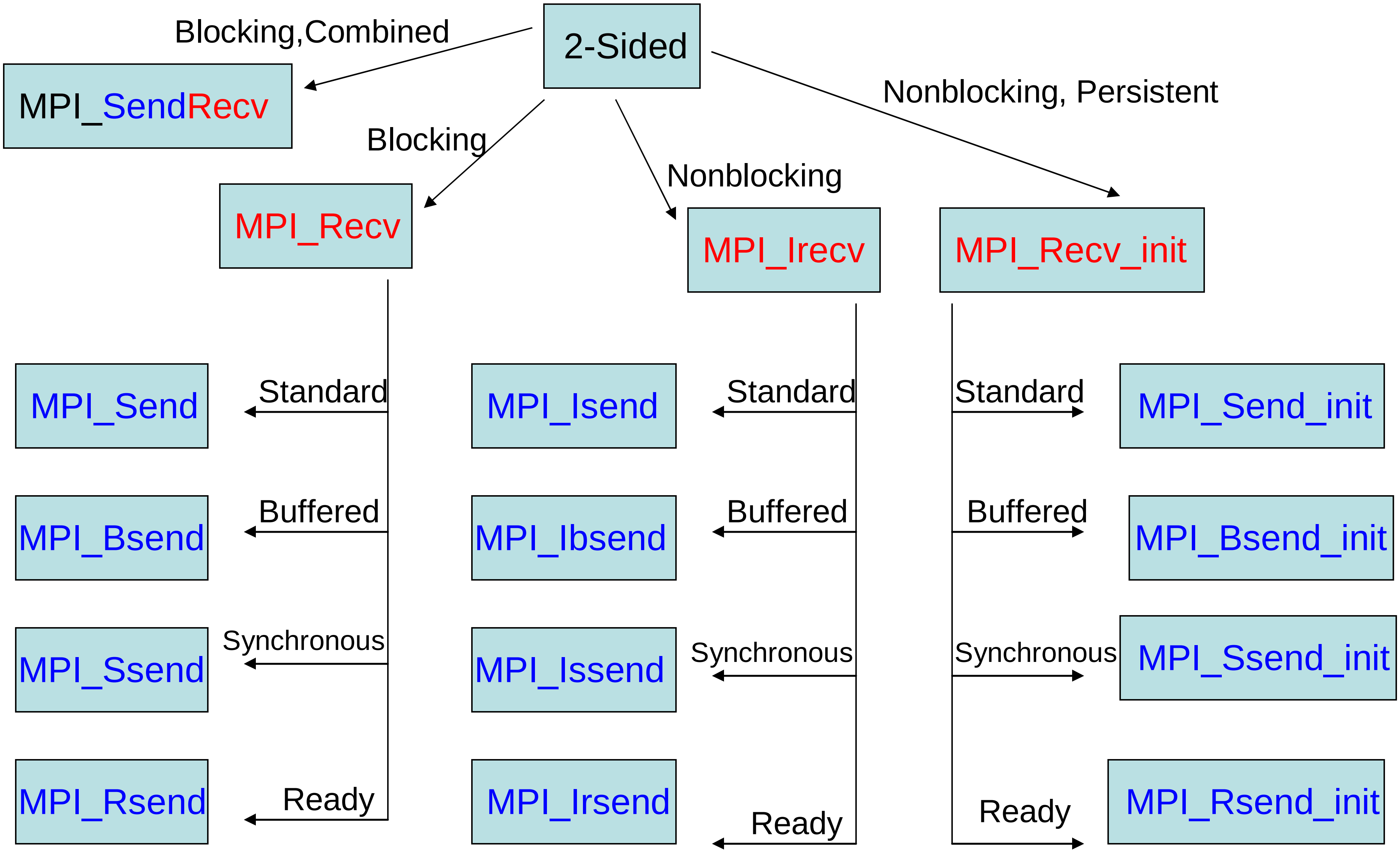
Communication vs. Computation



Improving Performance of Tau3P

- Performance greatly improved by better mesh partitioning
  - Previous work by Wolf, Folwell, Devine, and Pinar
- Possible improvements in scaled matrix/vector multiplication with vector addition algorithm
  - Different MPI communication methods
  - Different algorithm stage orderings
  - Thread algorithm stages

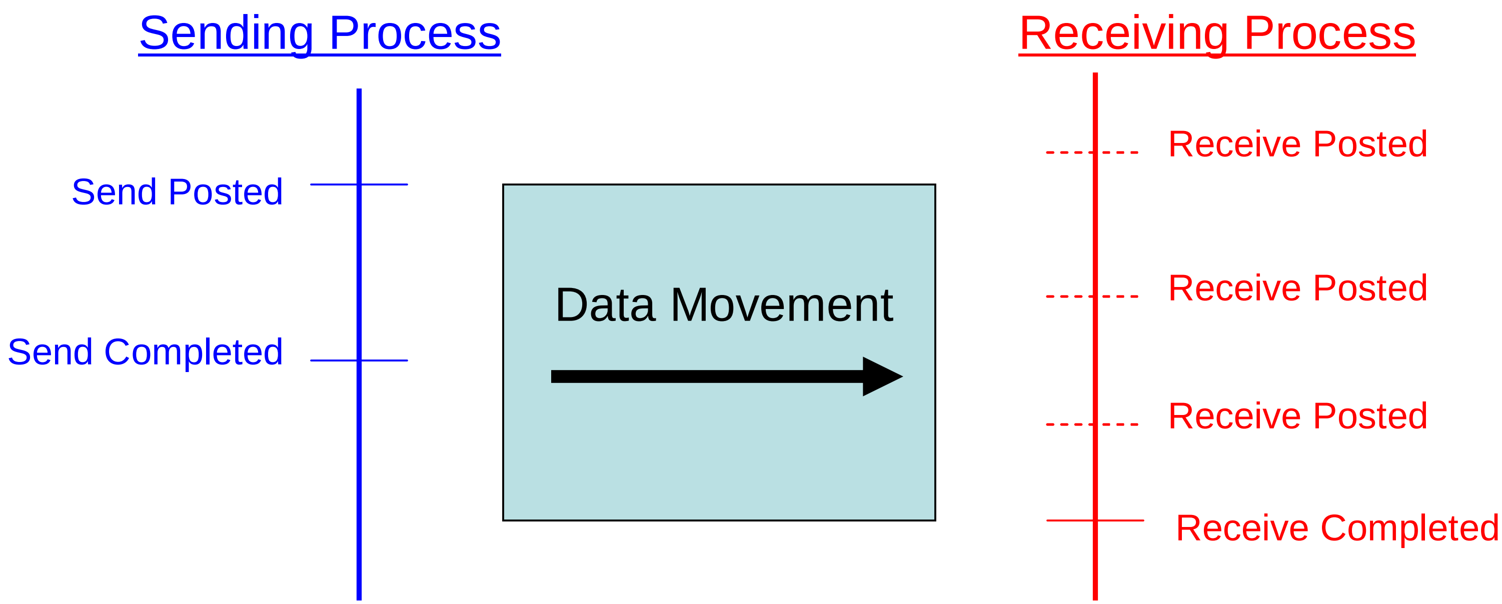
MPI 2-Sided Communication



Blocking vs. Nonblocking Communication

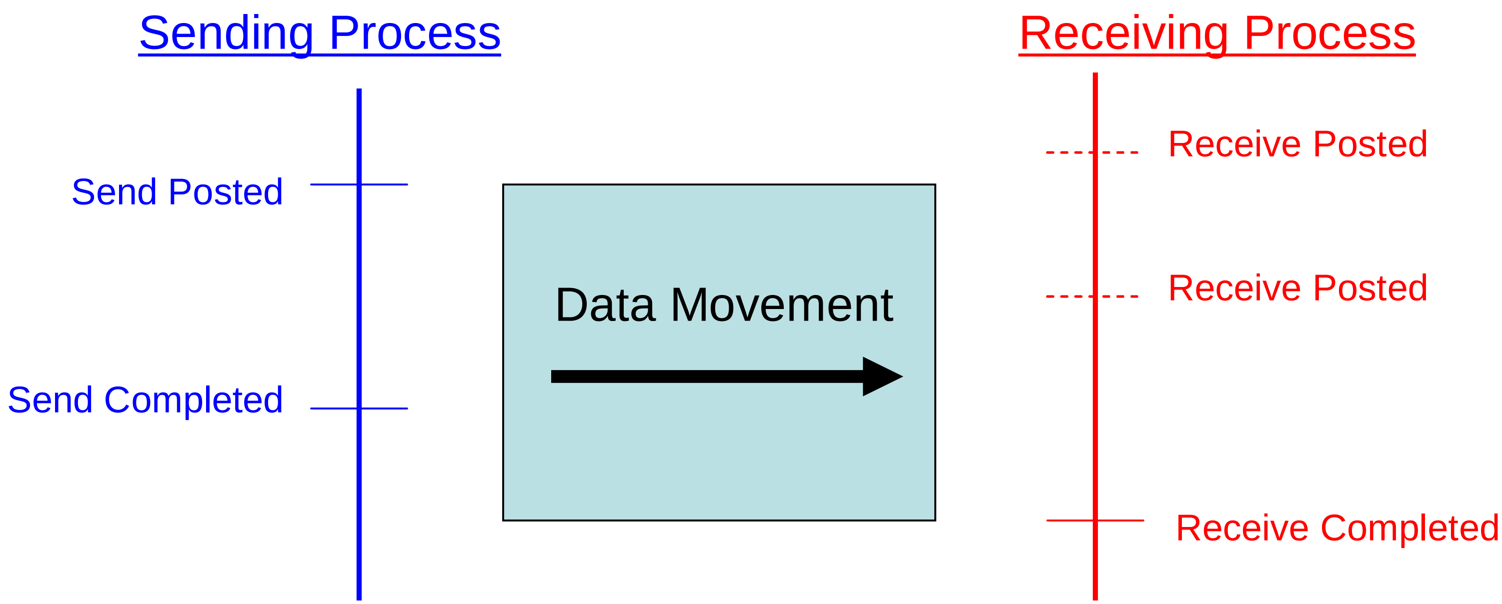
- Blocking
  - Resources can be safely used after return of call
  - MPI_Recv does not return until mesg received
  - Send behavior depends on mode
- Nonblocking
  - Resources cannot be safely used after return
  - MPI_Irecv returns immediately
  - Enables overlapping of communication with other operations
  - Additional overhead required
  - Used with MPI_Wait, MPI_Wait{all,any,some}, MPI_Test*
- Blocking sends can be used with nonblocking receives and vice versa.

Buffered Communication Mode



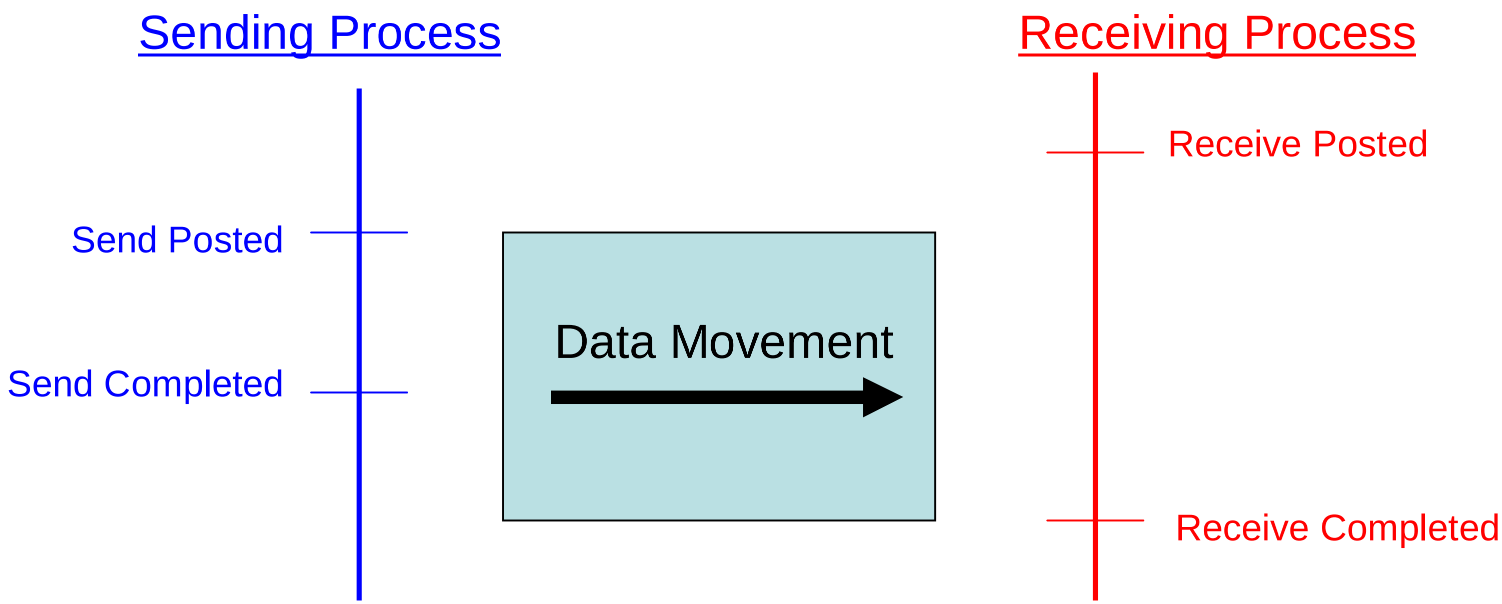
- MPI_Bsend, MPI_Ibsend
- A user defined buffer is explicitly attached using MPI_Buffer_attach
- Send posting/completion independent of receive posting

Synchronous Communication Mode



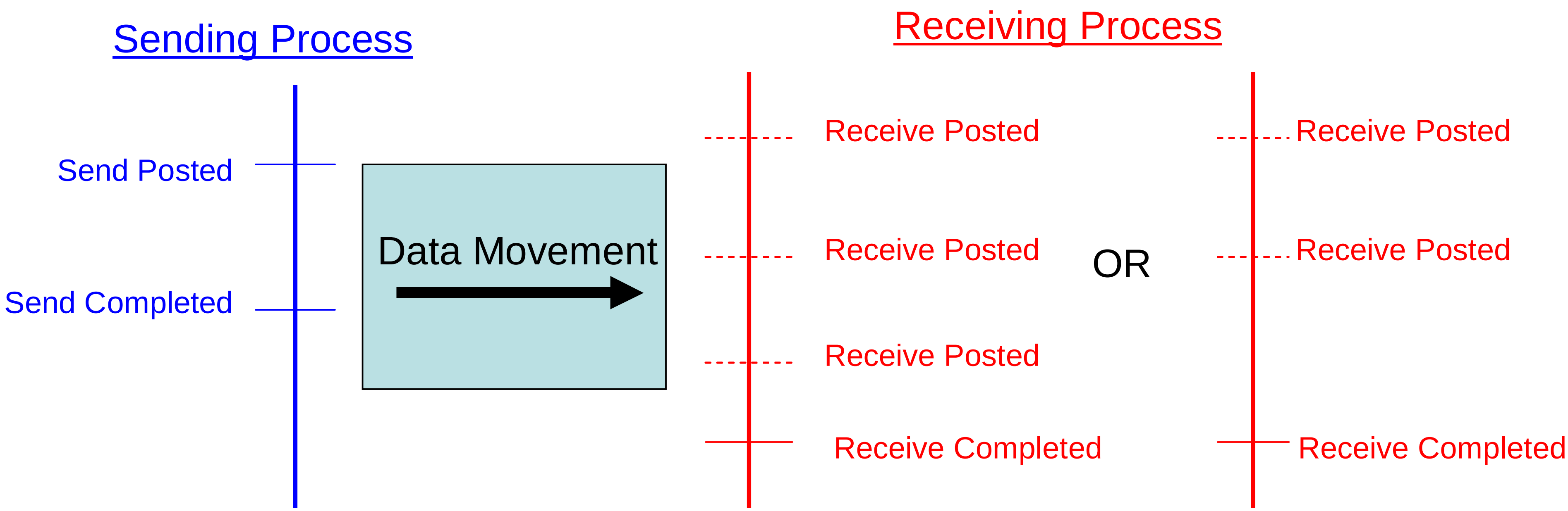
- MPI_Ssend, MPI_Issend
- Send can be posted independent of receive posting
- Send completion requires receive posting

Ready Communication Mode



- MPI_Rsend, MPI_Irsend
- Send posting requires receive to be already posted

Standard Mode Send



- MPI_Send, MPI_Isend
- Behavior is implementation dependent
- Can act either as buffered (system buffer) or synchronous

Persistent Communication

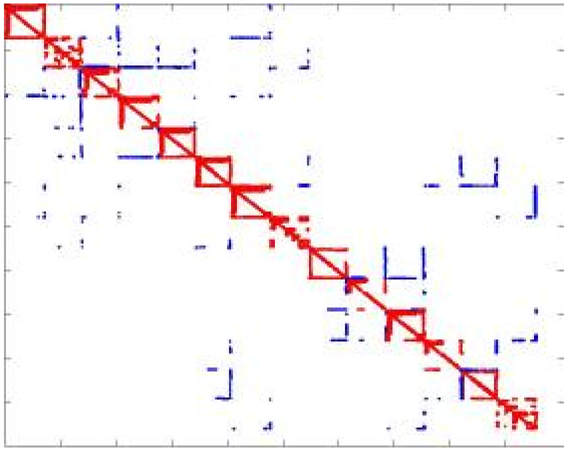
- Used when communication function with same arguments repeatedly called
- Bind list of communication arguments to persistent communication request
- Potentially can reduce communication overhead
- Nonblocking
- Argument list is bound using
  - MPI_Send_init, MPI_Bsend_init, MPI_Ssend_init, MPI_Rsend_init
- Request is initiated using MPI_Start

Basic Algorithm

- Scaled Matrix/Vector Multiplication with vector addition

$$\vec{e} += \alpha \cdot A_H \cdot \vec{h}$$

- Row partitioning of matrix and vector so all nonlocal operations are due to matrix/vector multiplication
- 3 main stages of algorithm
  - Multiplication and summation with local nonzeros
  - Communication of remote vector elements corresponding to nonlocal nonzeros
  - Remote multiplication and summation with nonlocal nonzeros



Implementation

- 42 different algorithms implemented
  - 3 Blocking send/recv algorithms
  - 3 Nonblocking send/recv algorithms
  - 36 Blocking send/nonblocking recv algorithms
    - 6 different orderings
    - Standard, buffered, synchronous, persistent standard, persistent buffered, persistent synchronous

Blocking Send/Blocking Receive Algorithms

Ordering	Stage 1	Stage 2	Stage 3
1	LC	Comm/RC	
2	LC	Comm	RC
3	Comm	LC	RC

Sends: MPI_Send, MPI_Sendrecv
Recvs: MPI_Recv, MPI_Sendrecv

Nonblocking Send/Nonblocking Receive Algorithms

Ordering	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5
4	Recv	Send	LC	Wait	RC
5	Send/ Recv	LC	Wait	RC	
6	Recv	Send	Wait	LC	RC

Sends: MPI_Isend
Recvs: MPI_Irecv

Blocking Send/Nonblocking Receive

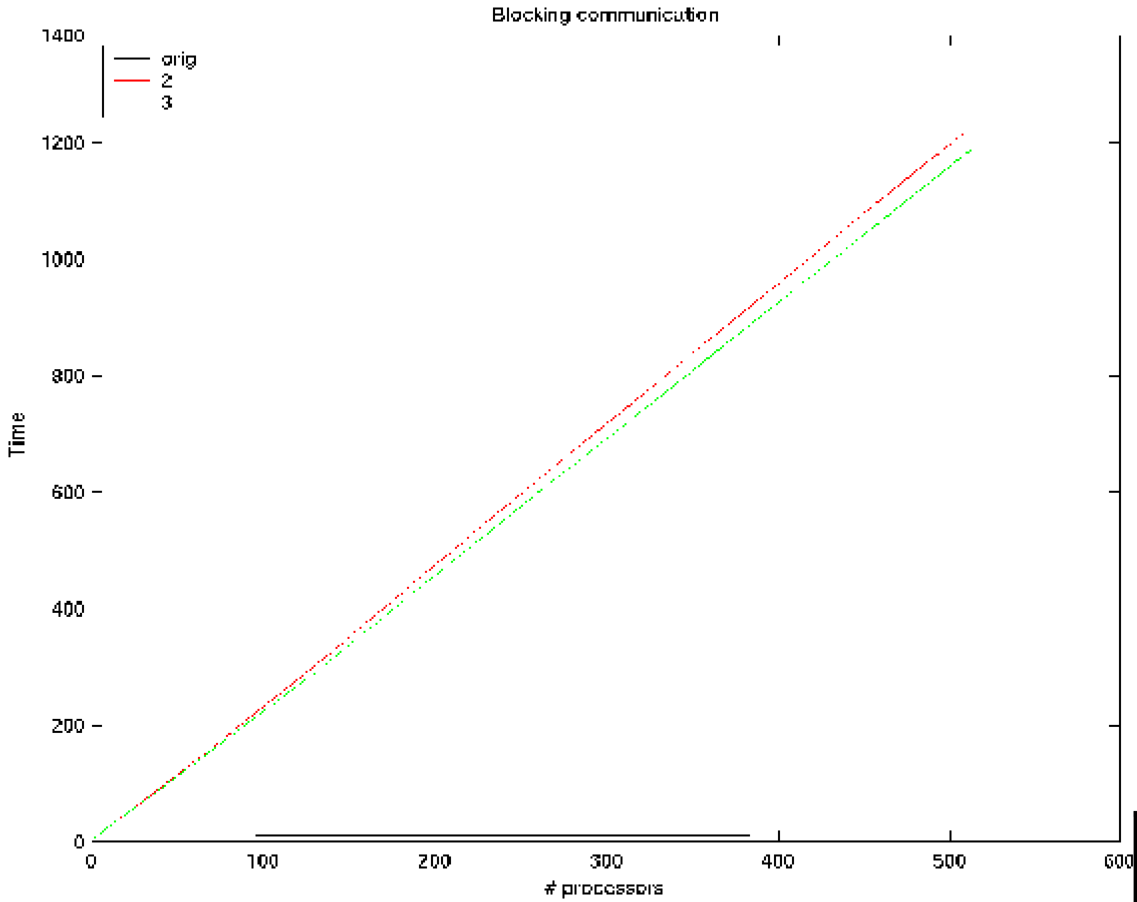
Ordering	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5
7	Recv	Send	Wait	LC	RC
8	Recv	Send	LC	Wait	RC
9	Recv	LC	Send/Waits	RC	
10	Recv	Sends/Waits	LC	RC	
11	Recv	LC	Send/Waits /RC		
12	Recv	Sends/Waits/RC	LC		

Sends: MPI_Send, MPI_Bsend,MPI_Ssend, MPI_Send_init,
MPI_Bsend_init, MPI_Ssend_init
Recvs: MPI_Irecv

Problem Setup

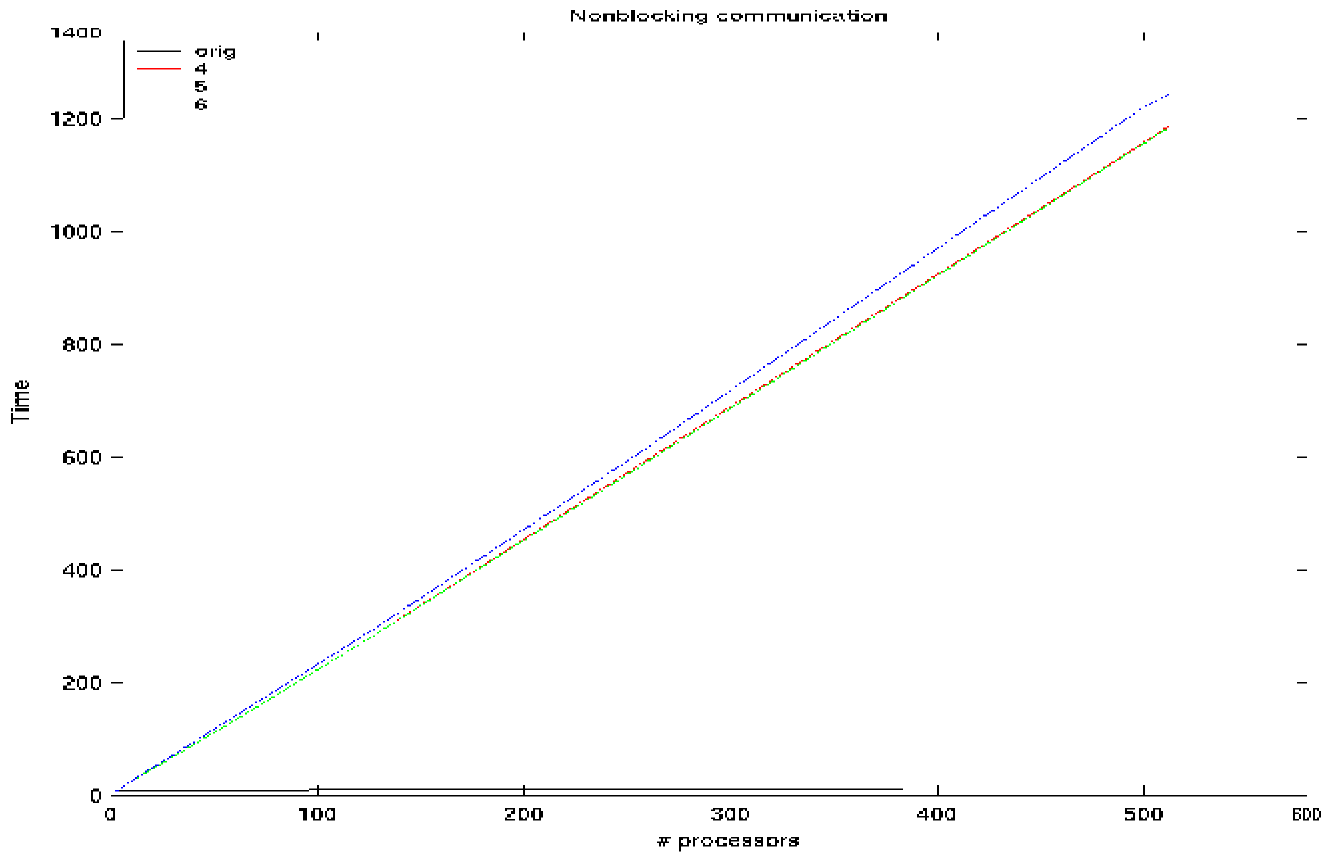
- Attempted to keep work per processor invariant
- Built nine 3D rectangular meshes using Cubit
  - External dimensions of meshes the same
  - Each mesh used for a particular number of processors (2, 4, 8, 16, 32, 64, 128, 256, 512)
  - Number of elements of each mesh controlled so that each matrix would have approximately 150,000 nonzeros per process for each mesh
- RCB 1D partitioning used
  - Meshes built to keep neighboring processors to minimum
- All runs on IBM SP at NERSC (seaborg)

Blocking Communication



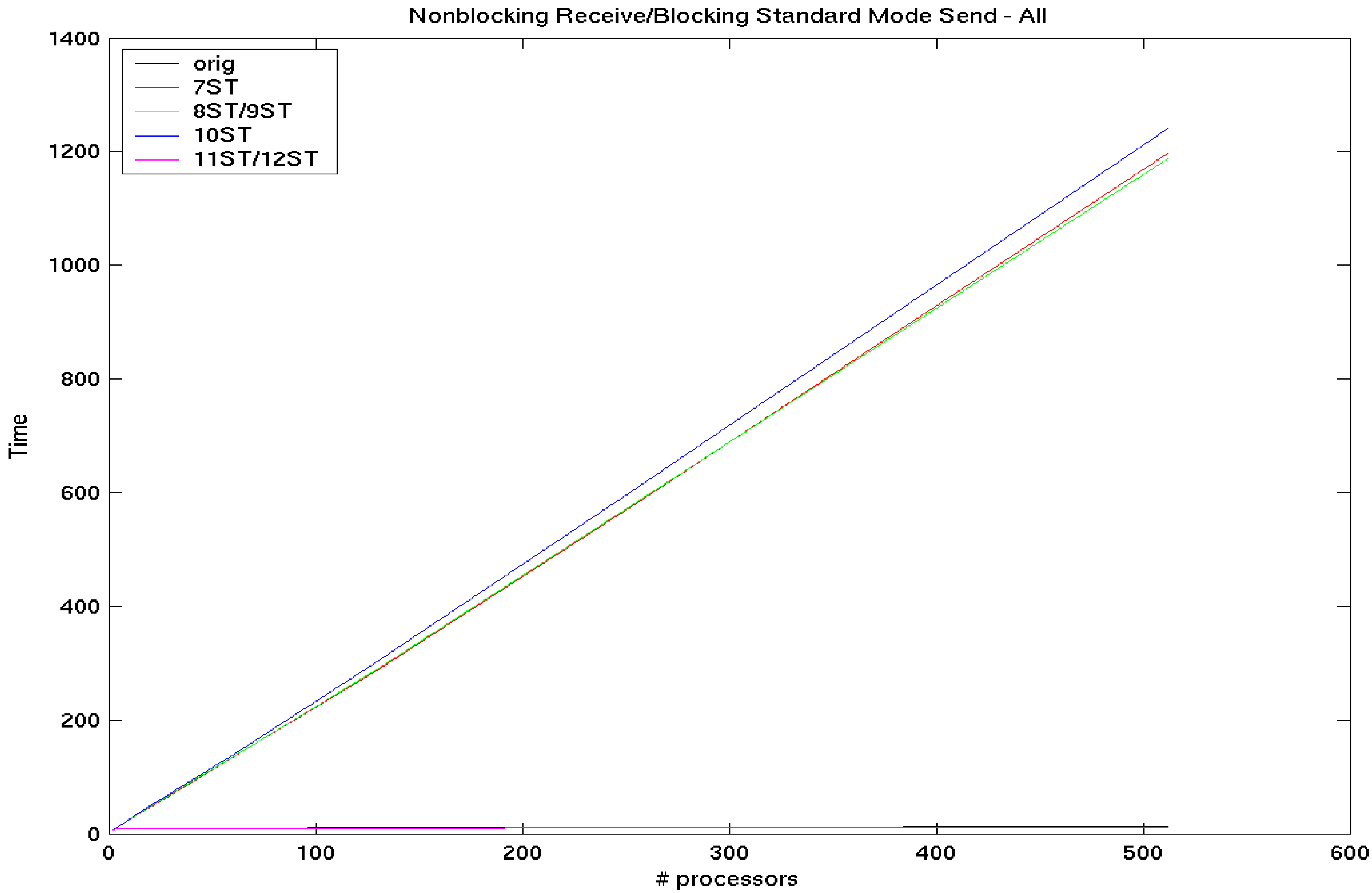
Method	Stage 1	Stage 2	Stage 3
orig	LC	Comm/RC	
2	LC	Comm	RC
3	Comm	LC	RC

Nonblocking Communication



	S1	S2	S3	S4	S5
Ord. 4	Recv	Send	LC	Wait	RC
Ord. 5	Send/Recv	LC	Wait	RC	
Ord. 6	Recv	Send	Wait	LC	RC

Blocking Send/Nonblocking Recv: Standard

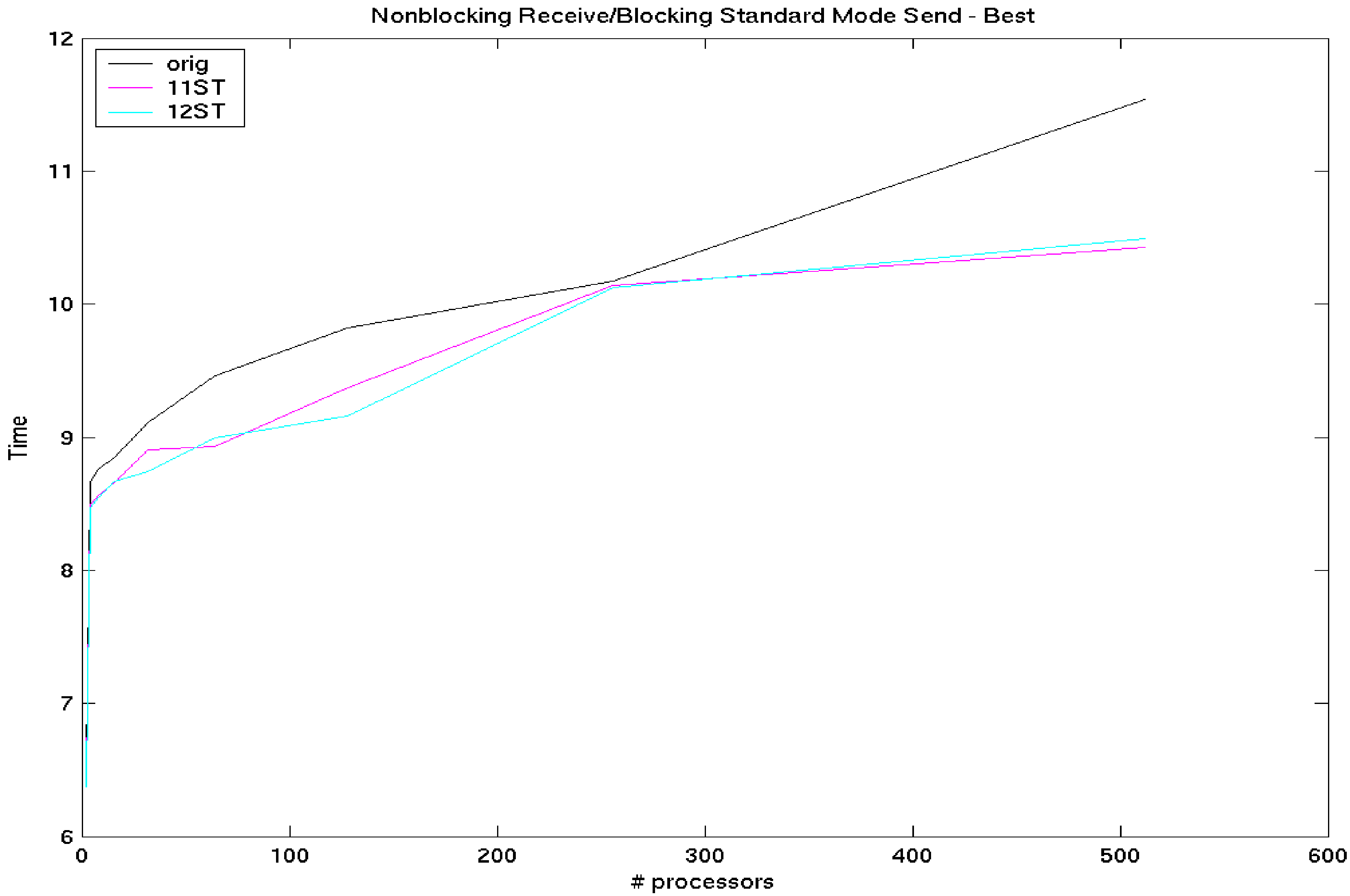


Orderings for Blocking Send/Nonblocking Recv

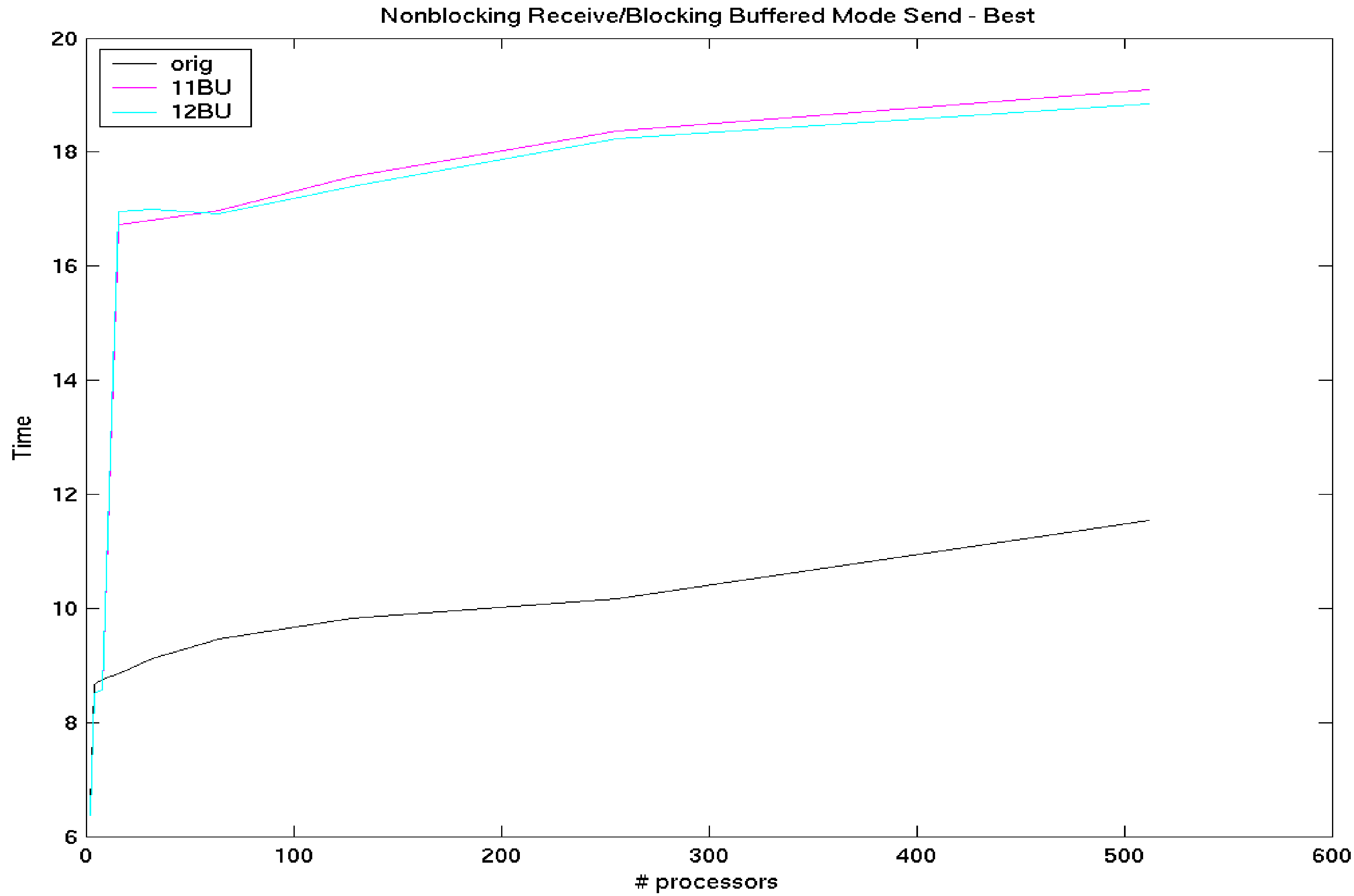
- Orderings 7-10 performed significantly worse than 11-12 for all MPI modes
- Subsequent graphs show only best algorithms

Ordering	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5
7	Recv	Send	Wait	LC	RC
8	Recv	Send	LC	Wait	RC
9	Recv	LC	Send/Waits	RC	
10	Recv	Sends/Waits	LC	RC	
11	Recv	LC	Send/Waits /RC		
12	Recv	Sends/Waits/RC	LC		

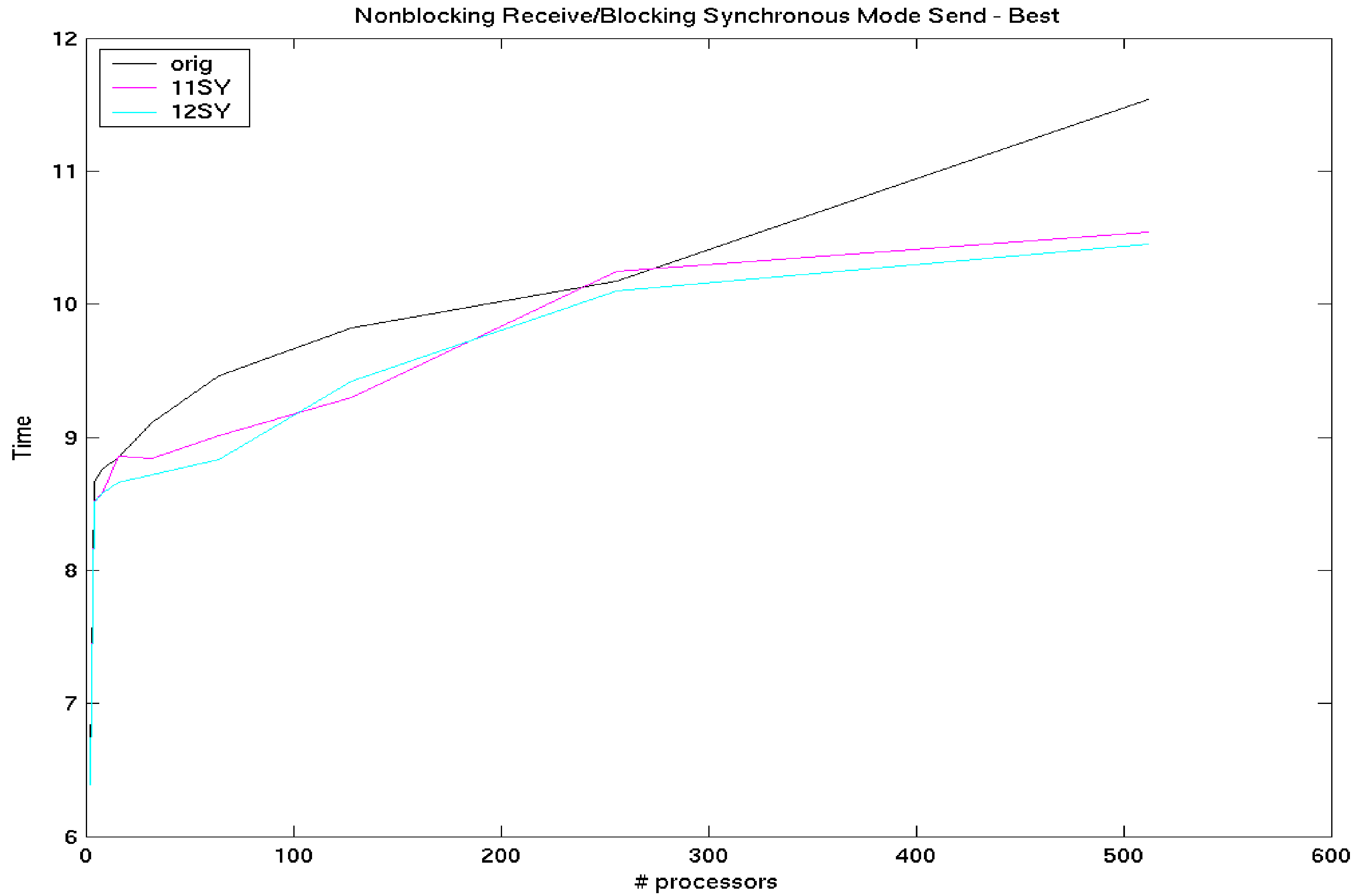
Blocking Send/Nonblocking Recv: Standard



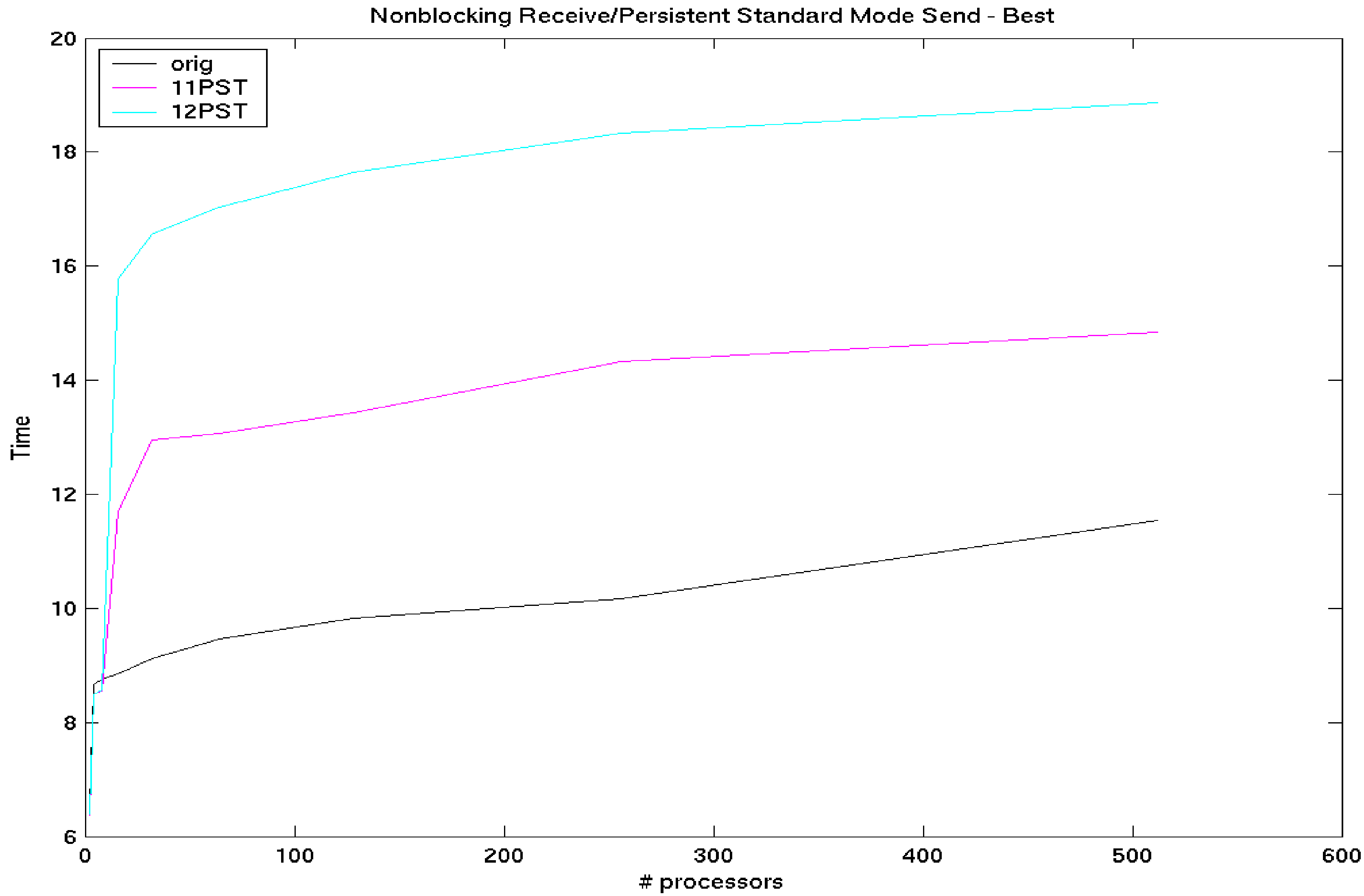
Blocking Send/Nonblocking Recv: Buffered



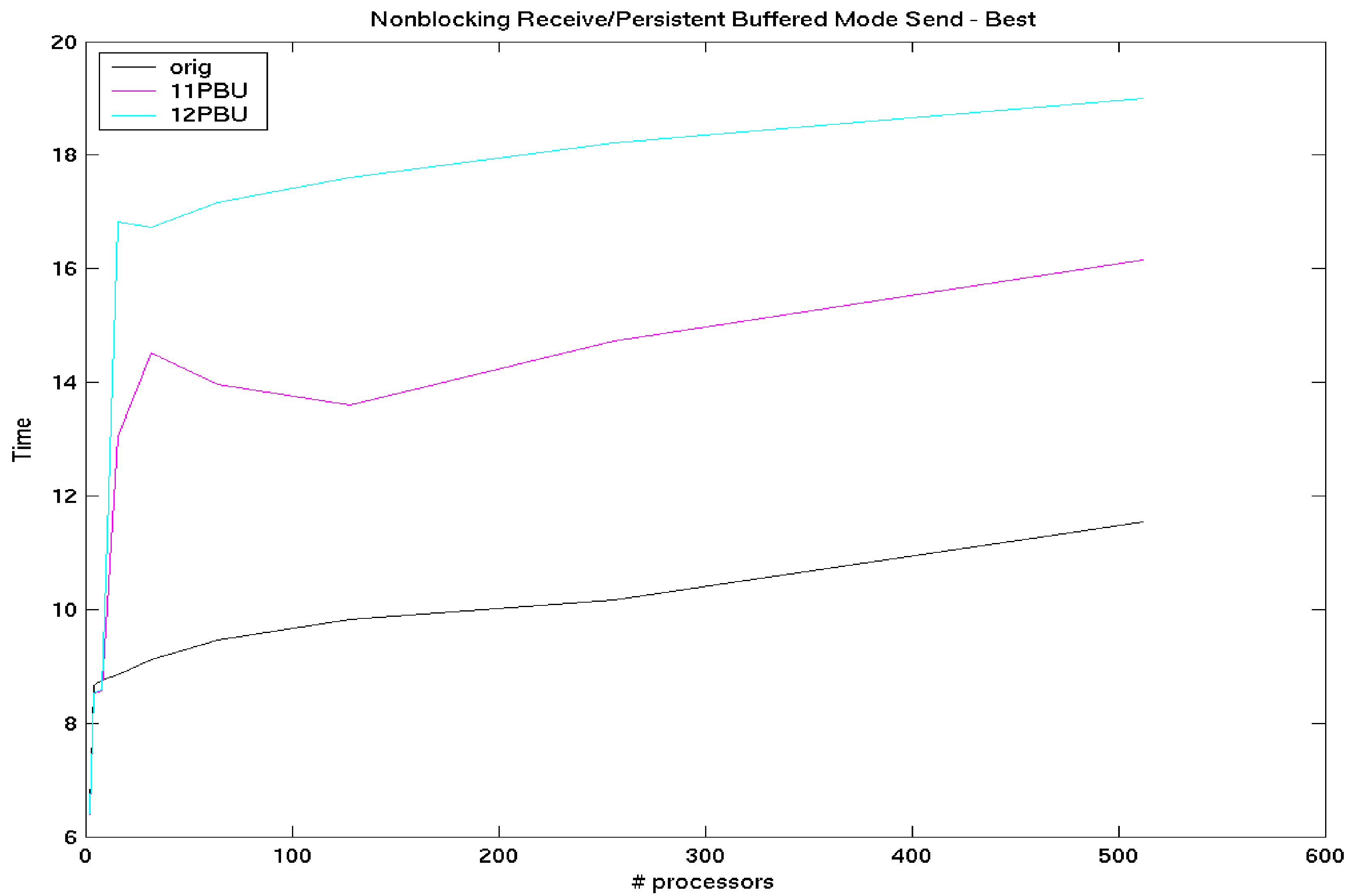
Blocking Send/Nonblocking Recv: Synchronous



Blocking Send/Nonblocking Recv: Persistent Standard

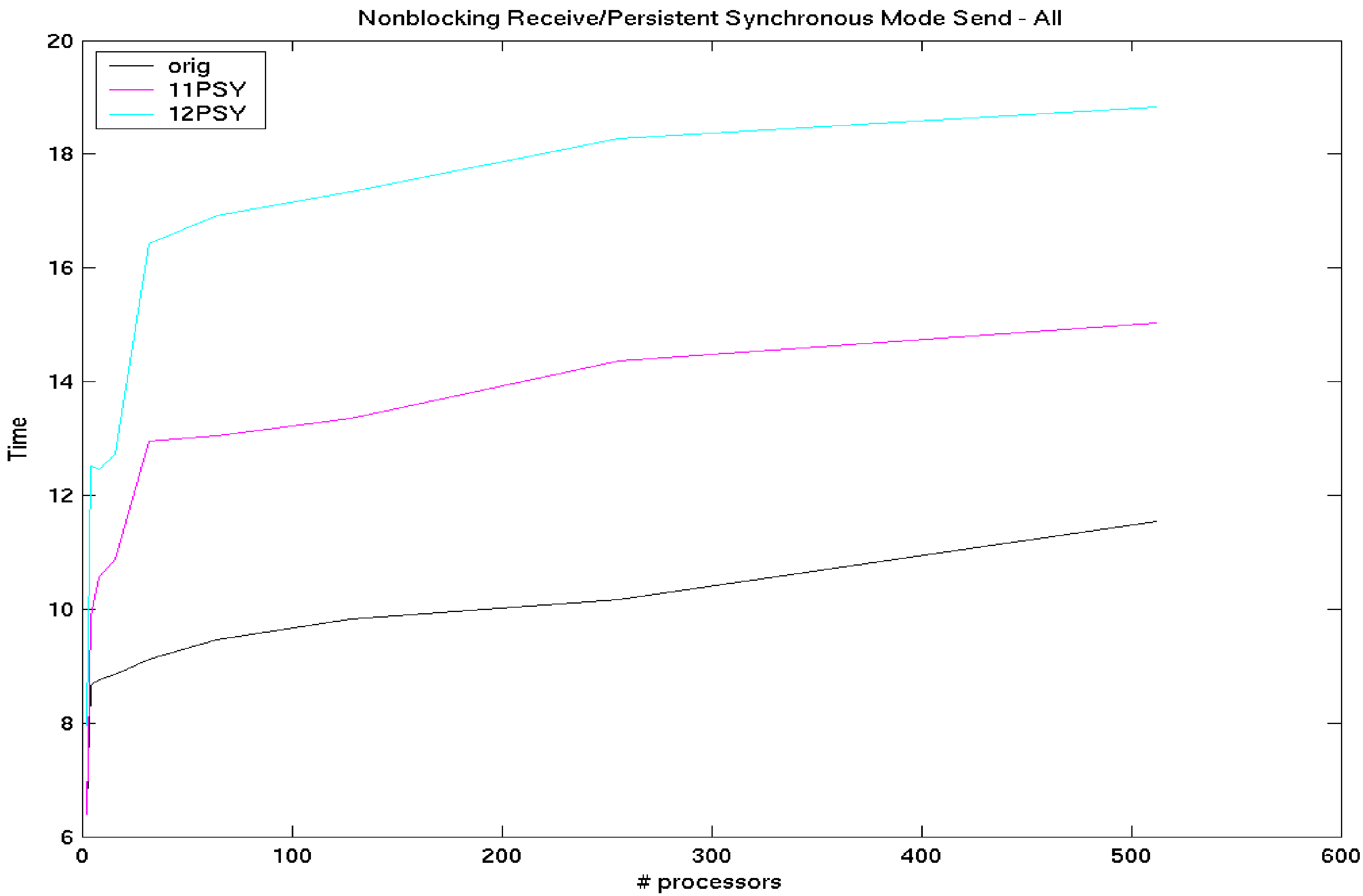


Blocking Send/Nonblocking Recv: Persistent Buffered

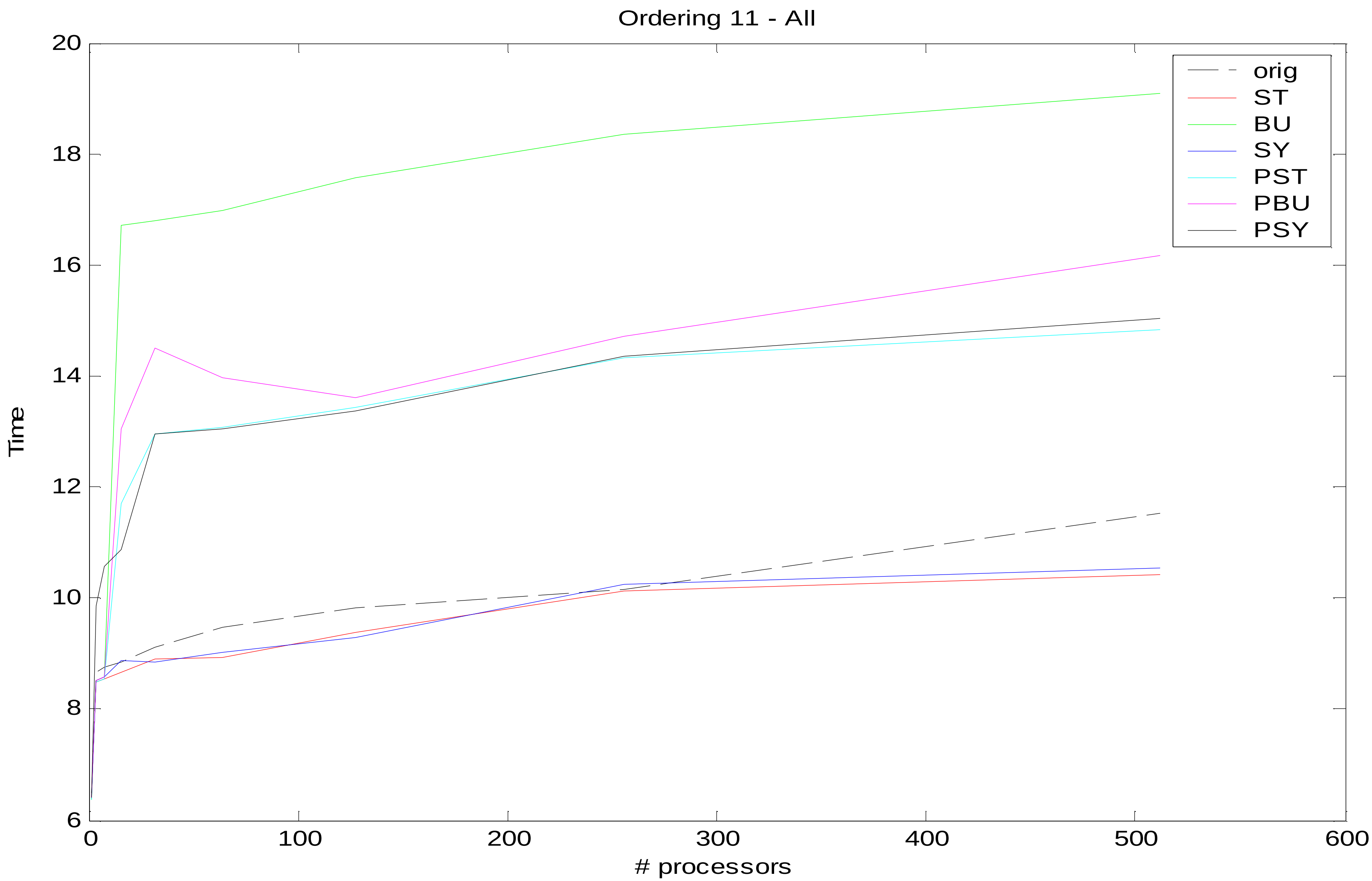


ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

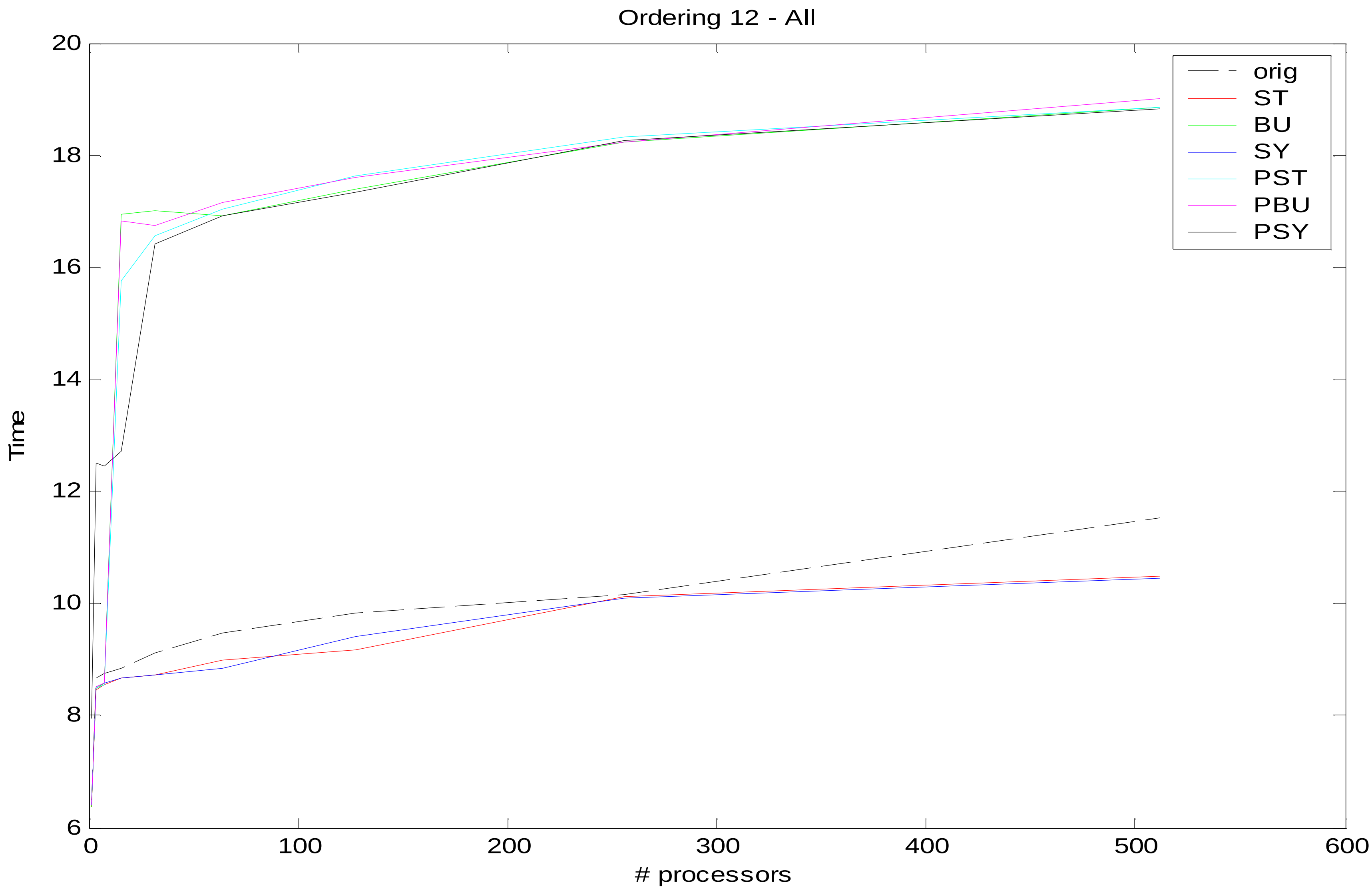
Blocking Send/Nonblocking Recv: Persistent Synchronous



Ordering 11

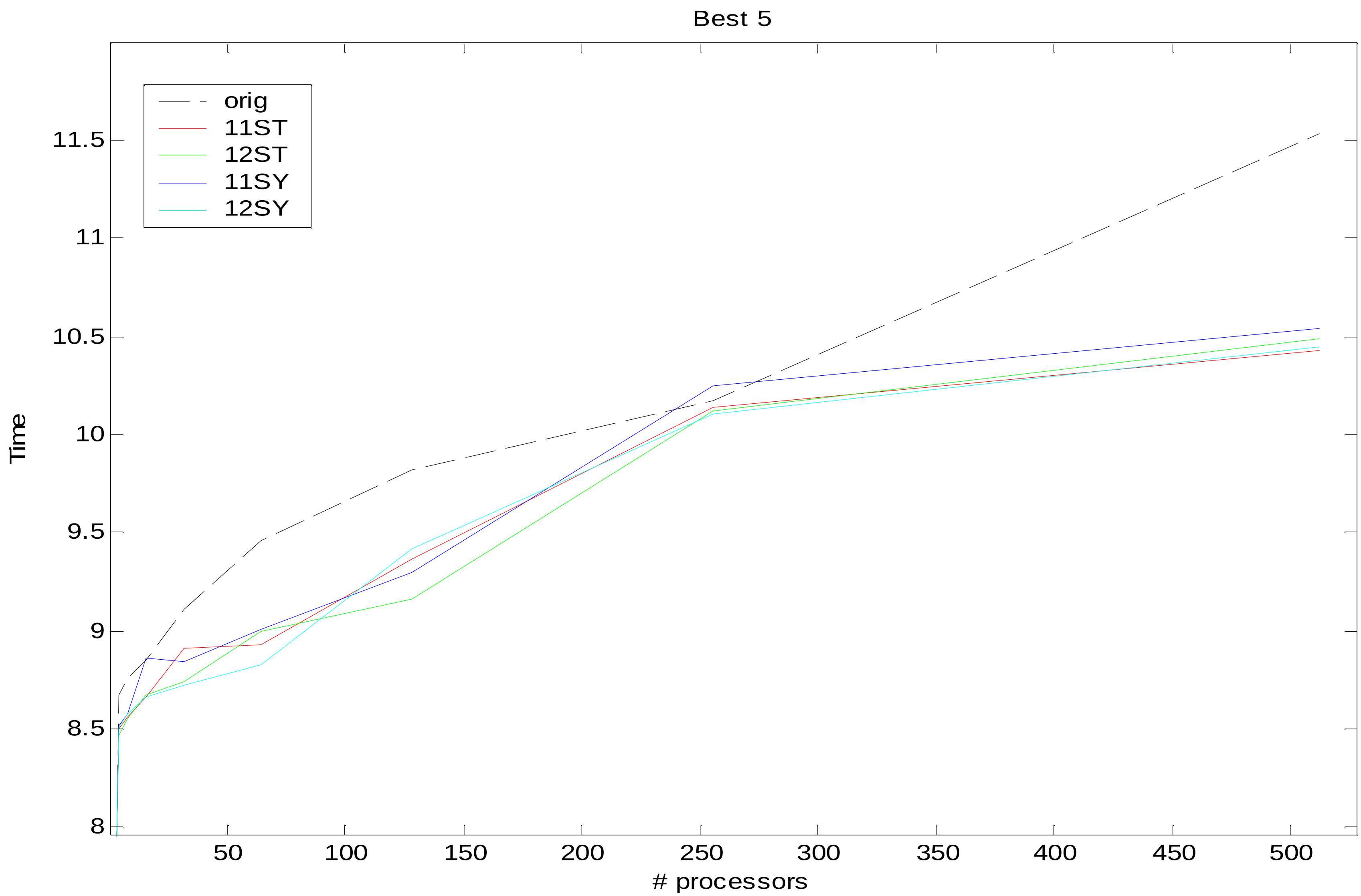

Ordering 11 - All
Time
processors
20
18
16
14
12
10
8
6
0
100
200
300
400
500
600
orig
ST
BU
SY
PST
PBU
PSY

Ordering 12



Ordering 12 - All
Time
processors
20
18
16
14
12
10
8
6
0
100
200
300
400
500
600
orig
ST
BU
SY
PST
PBU
PSY

Best 5


Best 5
orig
11ST
12ST
11SY
12SY
Time
processors
11.5
11
10.5
10
9.5
9
8.5
8
50
100
150
200
250
300
350
400
450
500

Observations/Conclusions

- Slight variation of algorithm can give very different performance
- Algorithm (with Tau3P data) very sensitive to stage ordering
- Combined communication/remote computation steps very beneficial
- Standard and Synchronous modes good
- Buffered modes costly
- Persistent communication costly
- Some factors could be machine dependent

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Future Work

- Threaded algorithms
  - Preliminary results not good
- Visualization of simulations
- Real accelerator structure
- Scalability of fixed size problem

Acknowledgements

- LBNL
  - Ali Pinar, Esmond Ng
- SLAC (ACD)
  - Adam Guetz, Cho Ng, Nate Folwell, Kwok Ko
- MPI
  - Prof. Heath's CS 554 Class Notes
  - Using MPI, Gropp, Lusk, Skjellum
  - MPI: The Complete Reference, Snir, et al.

- Work supported by U.S. DOE CSGF contract DE-FG02-97ER25308