

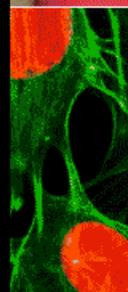


## The Cray X1E™ MVP Supercomputer

John M. Levesque

Director – Cray's Supercomputing Center  
of Excellence

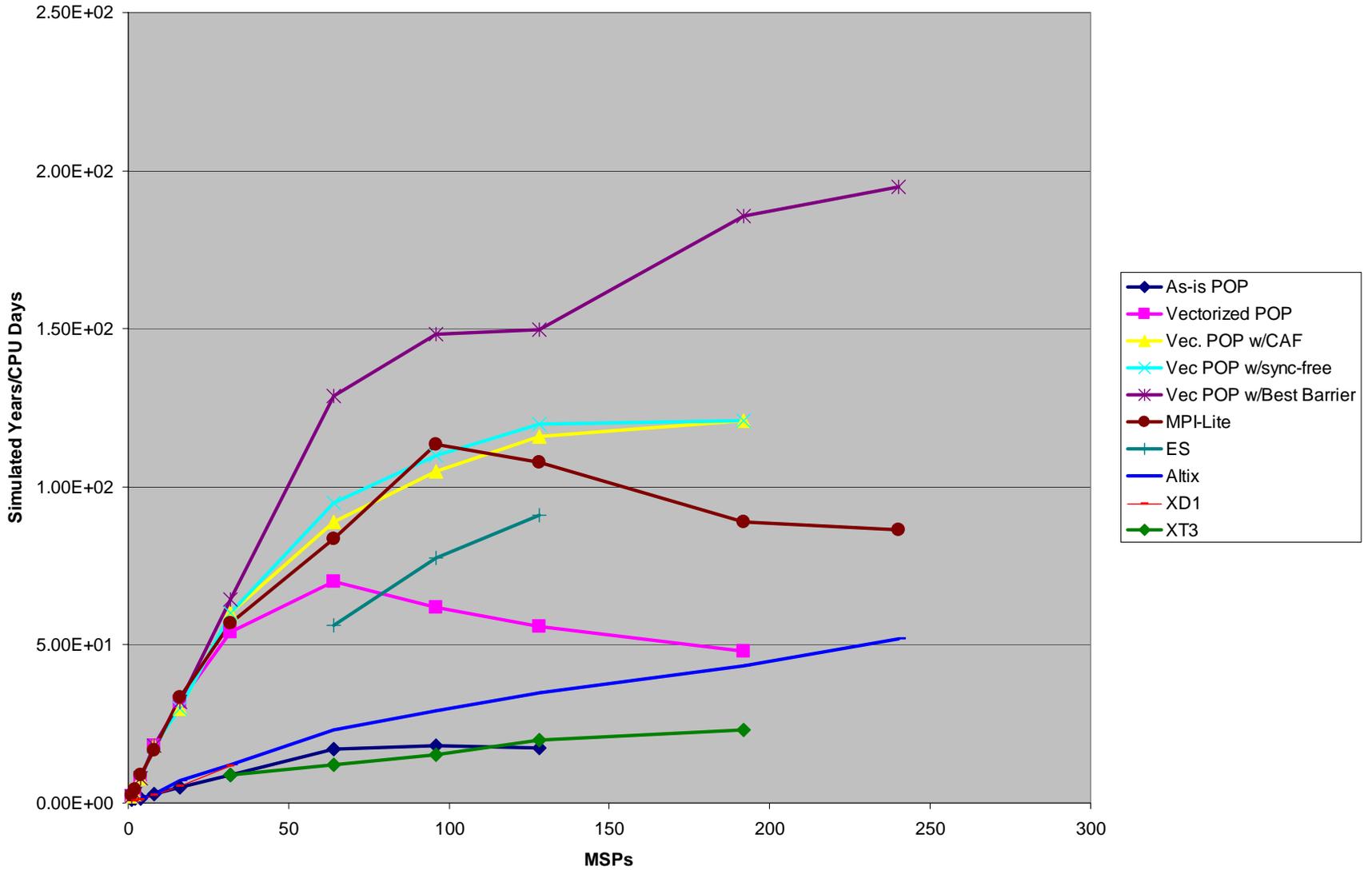
Oct, 2005



# X1E Programming Environment

- Use of CAF and SHMEM to optimize libraries that require messaging
- The PDGCS compiler is the best optimizing compiler in the world
- Luiz DeRose, the best tool developer in the world, will talk about his tools in the XT3 session

### Comparisons of Various Versions of POP



# Parallel Performance Optimization in LibSci

1. Locate communication bottlenecks
2. Use CAF or shmemp for data transfer
3. Utilize global addresses for optimal performance
4. Restructure parallel algorithms to utilize 1-sided communications

# 1. Communication bottlenecks

## 1. Algorithmic bottlenecks:

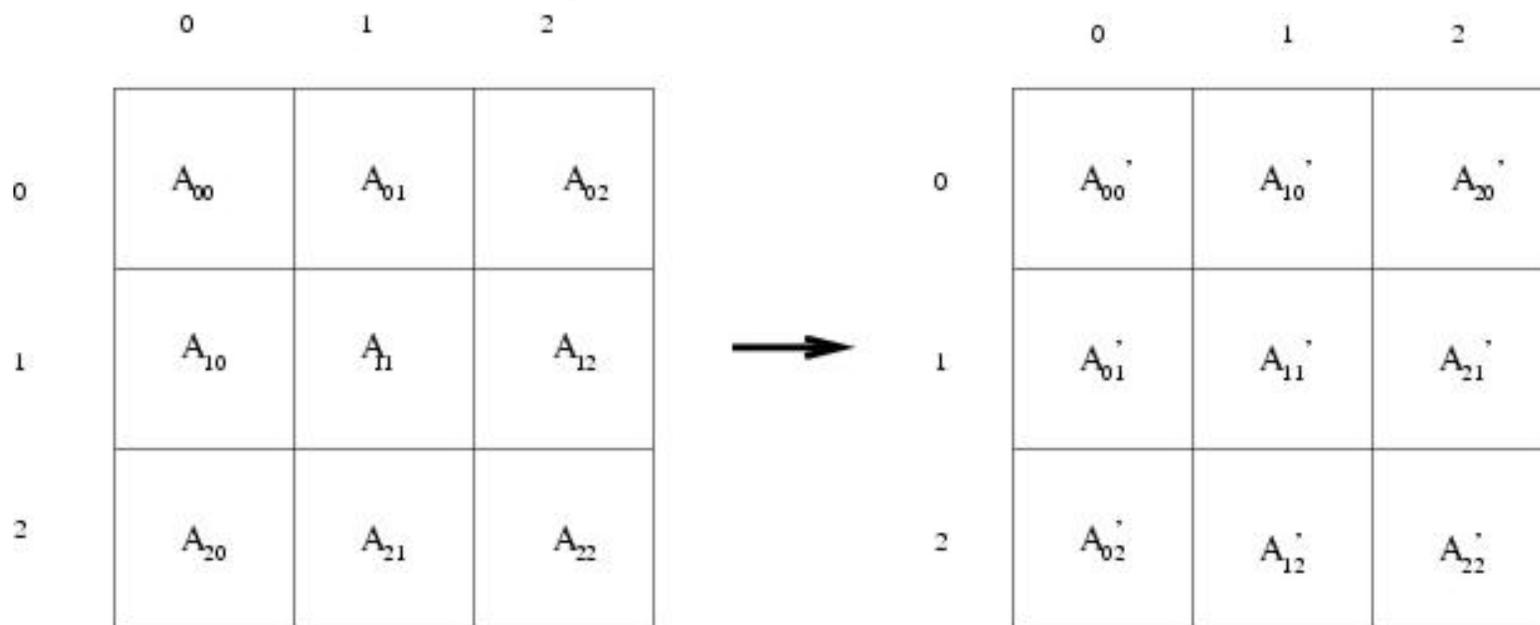
- Knowledge of the algorithm shows us that we expect heavy communications in a particular area
  - » We may or may not be able to reduce the communication cost

## 2. Architecture specific bottlenecks:

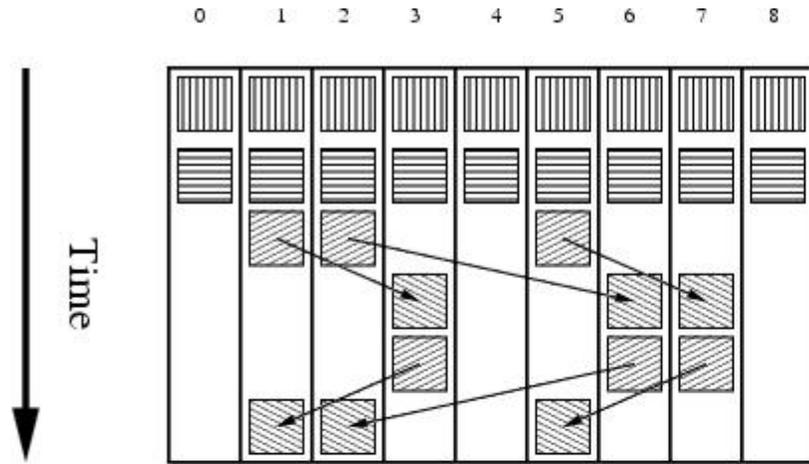
- The specifics of the architecture (e.g. MPI Implementation, interconnect bandwidth/latency, memory bandwidth, cache system) combine to give rise to areas heavy in communications.
  - » We can always reduce the communication cost

# 1-sided vs 2-sided

- For some data movement, shmem and CAF are better purely on the basis that they are 1-sided (so regardless of the lower latency etc)
- Consider for the following simple example : blocked matrix transpose on a 3x3 grid of processors

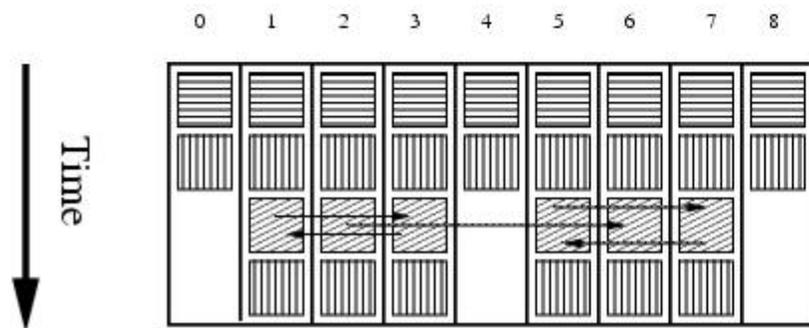


# One sided vs 2-sided



-  Barrier
-  Local transpose
-  MPI/Blacs Send
-  MPI/Blacs Recv

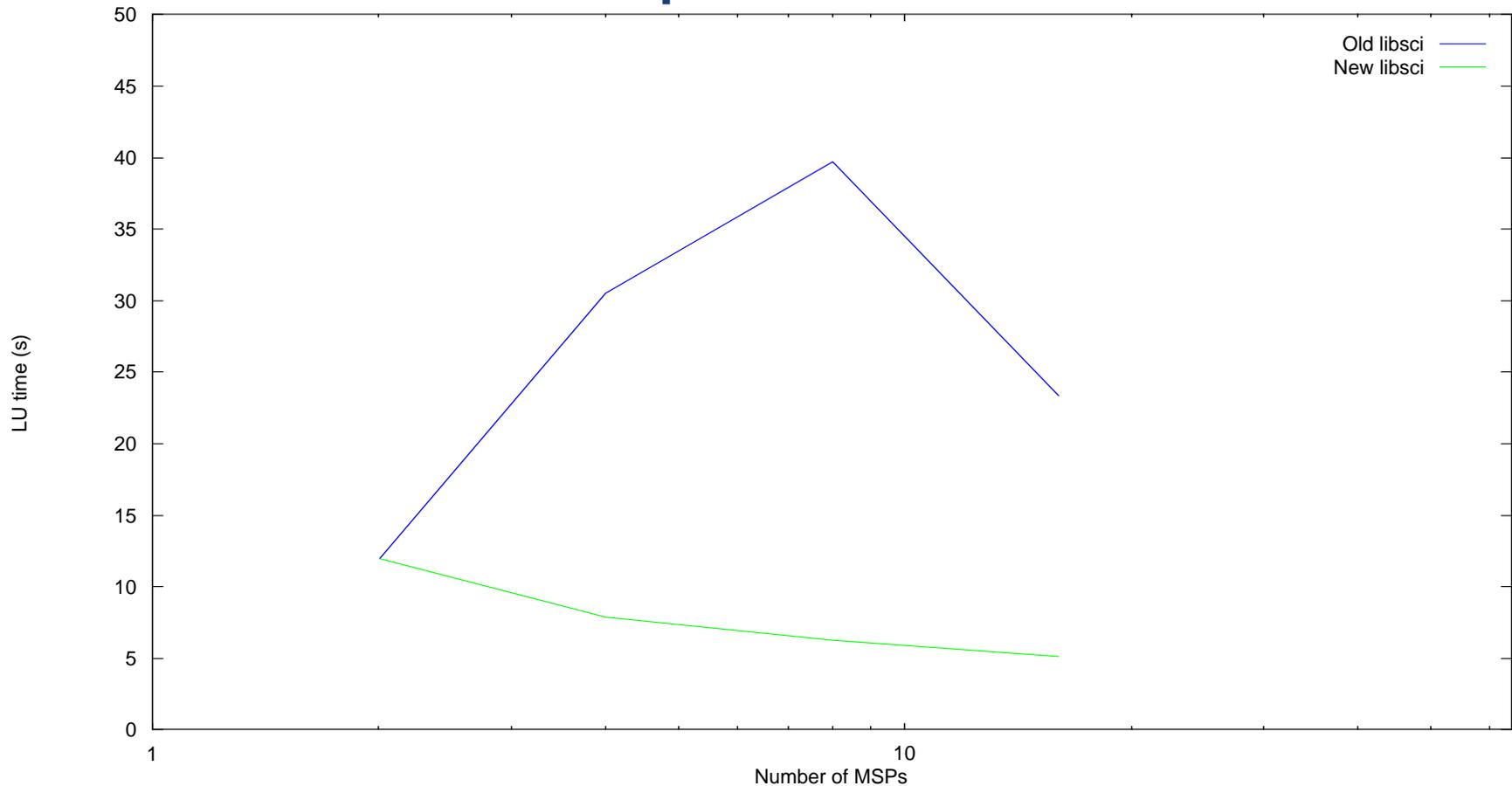
(a)



-  Local transpose
-  Barrier
-  Remote put (unanswered)

(b)

# Example 1. Communication bottlenecks removed in Scalapack PZGETRF



# Why not MPI-2 one-sided

- Really handshaking under the covers
- CAF is compiler assisted, just like a memory load/store to the compiler
- Hardware assist in the X1 – X1E

# The Best Optimizing Compiler in the World

- Best Vector compiler and now the best optimizing
  - According the the Nullstone Compiler suite
    - 47 Tests
      - As is compiled with optimization
      - Then manually restructured and compiled
      - Percent of optimization recorded



Nullstone Compiler Benchmark

