Implementing “Pliris-C/R” Into the EIGER Application

Mike Davis, Cray Inc.

CUG 2015

Agenda

- EIGER application
- Cielo system
- Pliris solver library
- Pliris-C/R
- Other resiliency features for EIGER
- Results from EIGER runs

EIGER

- **Frequency-domain EM code**
- **Dense matrix factor/solve, complex-valued elements**
  - Over 2M unknowns
  - Runs on 5000 Cielo (XE6) nodes, MPI everywhere
  - Factor takes ~80000 seconds

Cielo

- **96 cabinet XE6**
- **8944 compute nodes**
  - Dual-socket 8-core Opteron (Magny-Cours) 2.4GHz
  - 32 GB RAM
- **1.11 PF HPL**
  - Number 6 on TOP 500, June 2011

Pliris

- Dense solver package, part of Trilinos
- Block data distribution with torus-wrap mapping
- Block-cyclic work distribution (LU decomposition)
- Shuffle permutation of solution
- RHS vectors known in advance

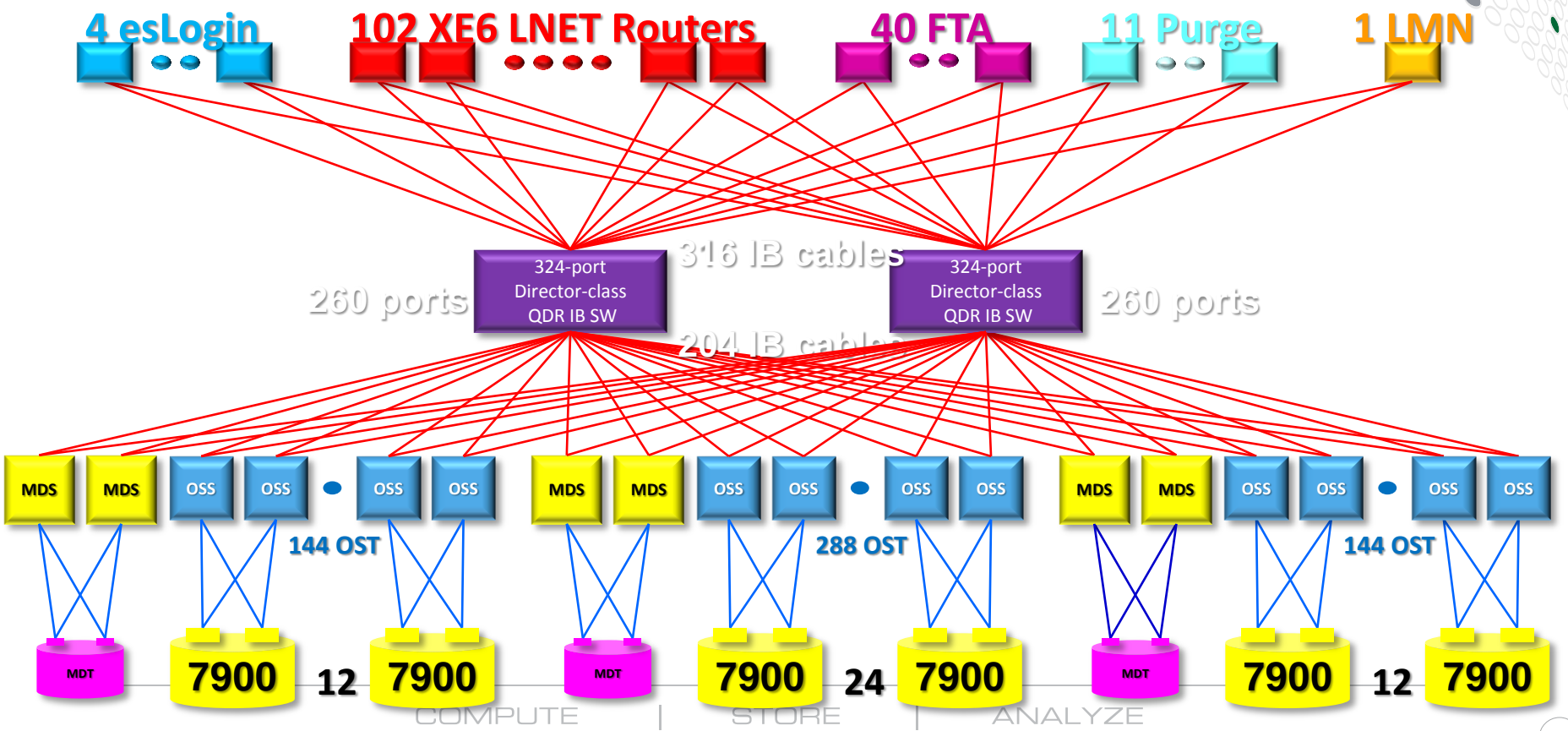
Pliris-C/R Design

- **Checkpoint/restart covers only factor()**
  - Checkpoint occurs inside loop over columns
  - Restart occurs above loop over columns
- **Process checkpoint image includes:**
  - Local block of matrix (>1 GB/process)
  - Only relevant fraction of operand matrix saved
  - Work vectors
  - Pointers

Pliris C/R Design (2)

- Every process does I/O (no aggregation)
- I/O operations are POSIX unbuffered
  - preadv(), pwritev()
- Checkpoint files spread across multiple Lustre file systems
- N processes ←→ M files, with turnstiling
- Checkpoint operations spaced evenly across factor() column loop work space

Cielo esFS Configuration

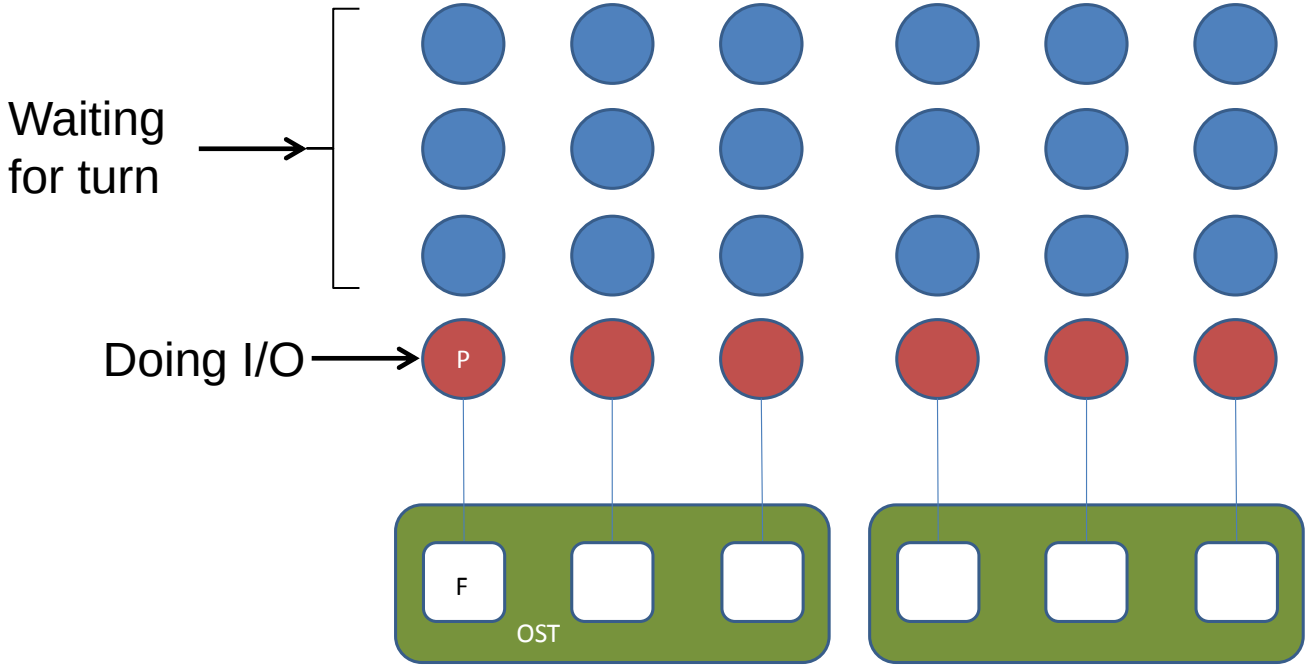


Cielo /lscratch3 I/O Bandwidth (MiB/sec)

- **N processes → N files using LANL fs_test**
  - Source: B.M. Kettering, CUG 2014 Proceedings

Processes	Eff. BW	Raw BW	
1024	73900	74400	
2048	77400	78500	Optimum
4096	76200	75500	
8192	72000	75900	
16384	64000	72000	
32768	57600	69400	
65536	43600	60900	

Turnstiling Basics



Turnstiling Optimizations

- **Processes that share a node take turns**
  - Keeps injection demand below limit
- **Processes sharing an OS image share open file descriptors**
  - Reduces metadata load
  - (Source: W. R. Stevens, “Advanced Programming in the UNIX® Environment”, 1993)

Single-OST Checkpoint Times (sec)

Test	Avg	Std Dev
NXN	11640	367
NX1	7697	721
NX5	7747	697
TURN5	6918	800
TURN5_SFD	6718	665

Pliris-C/R Tuning Parameters

- PLIRIS_CR_NFS: number of file systems
- PLIRIS_CR_DIR: directory paths; 1 per FS
- PLIRIS_CR_NS: OST counts; 1 per FS
- PLIRIS_CR_NF: number of files in checkpoint set
- PLIRIS_CR_COUNT: number of checkpoint sets to write over the course of factor()
- PLIRIS_CR_SIGNUM: signal number for imminent termination due to wall time or scheduled shutdown

Pliris-C/R Settings for EIGER

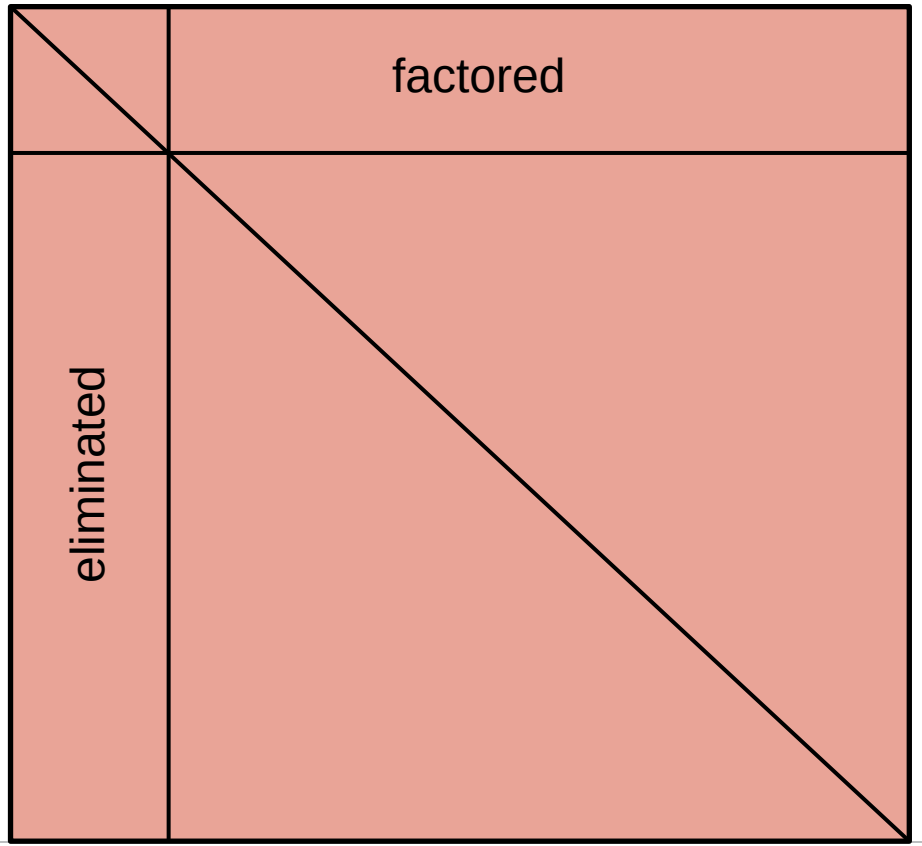
- PLIRIS_CR_NFS=3
- DIR2=/lscratch2/${USER}/${PBS_JOBNAME}
- DIR3=/lscratch3/${USER}/${PBS_JOBNAME}
- DIR4=/lscratch4/${USER}/${PBS_JOBNAME}
- PLIRIS_CR_DIR="${DIR2} ${DIR3} ${DIR4}"
- PLIRIS_CR_NS="125 250 125"
- PLIRIS_CR_NF=2500
- PLIRIS_CR_COUNT=6
- PLIRIS_CR_SIGNUM=23

Coordination of Checkpoints

- Selected iterations of loop over columns
- $b_i = N - \sqrt[3]{k+1-i} \left[\frac{N}{\sqrt[3]{k+1}} \right]$
- i is the checkpoint number (1 .. k)
- b_i is the column index at which checkpoint i is written
- N is the size of the matrix (trip count of column loop)
- k is the number of checkpoints to write (PLIRIS_CR_COUNT)

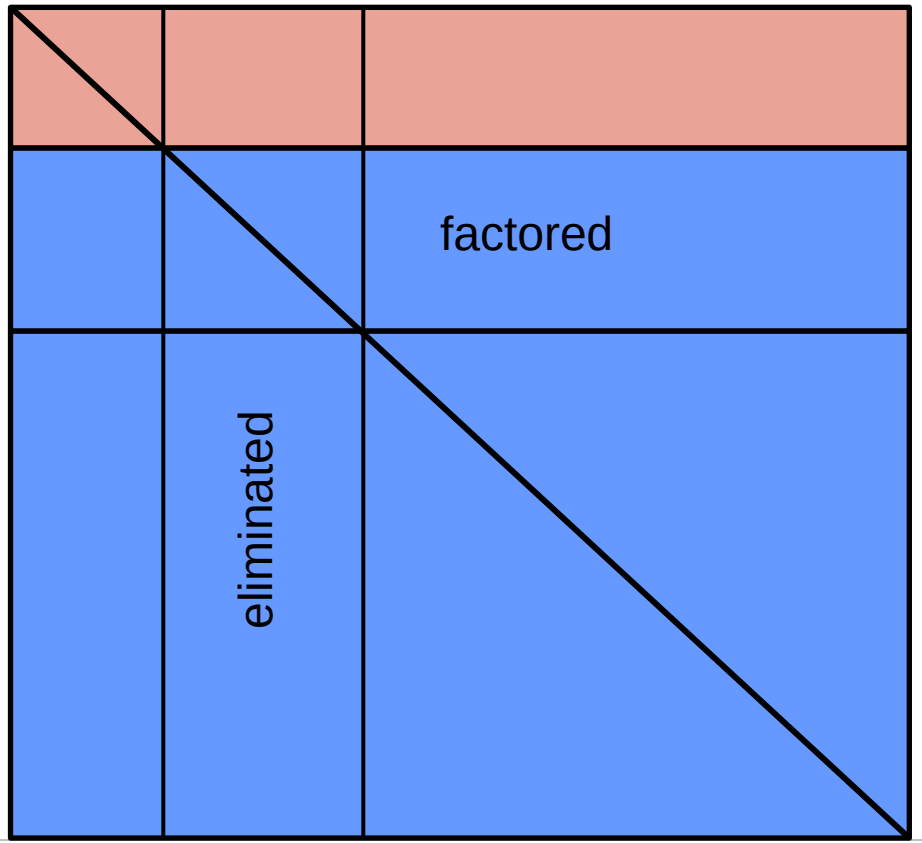
Decrementing Checkpoint of Matrix

$$E_i = \frac{Nb_{i-1}}{p^2}$$



Decrementing Checkpoint of Matrix (2)

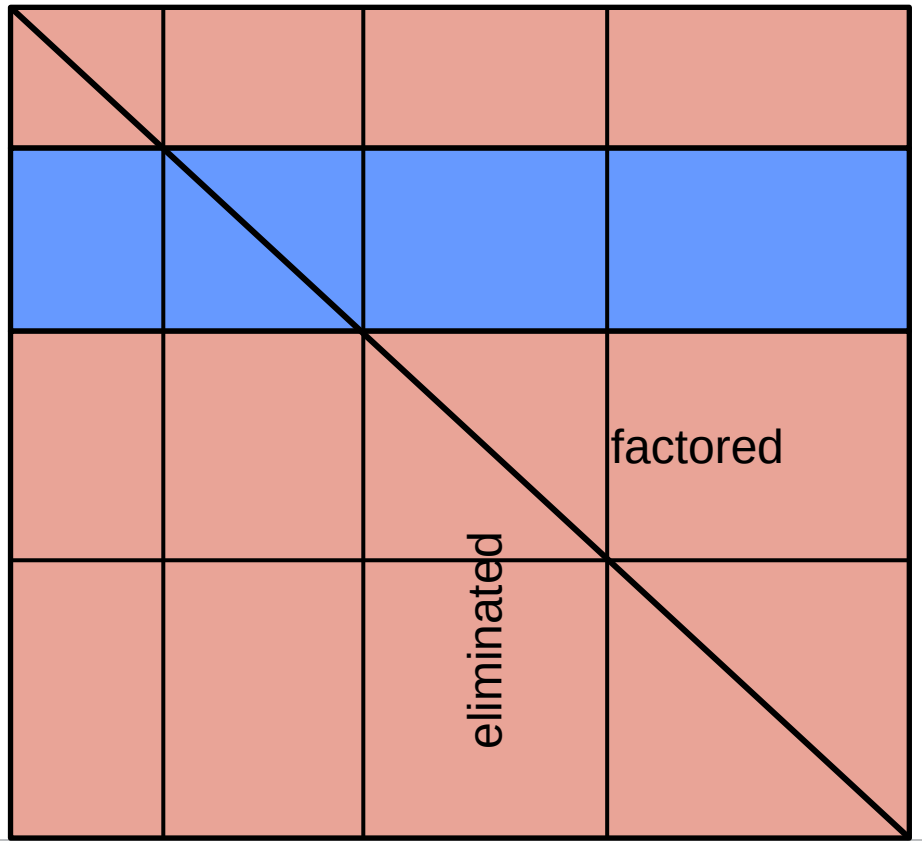
$$E_i = \frac{Nb_{i-1}}{p^2}$$



E_1
E_2
E_3
E_4
N^2/p^2

Decrementing Checkpoint of Matrix (3)

$$E_i = \frac{Nb_{i-1}}{p^2}$$



Selection of Checkpoint Count

- Minimize total work time
  - Source: J.T. Daly, Future Generation Computer Systems, Vol. 22, 2006
- $T_W(N) = M * e^{(F+\rho)/M} * \sum_{i=1}^{N}\left(e^{(T_S/N+\delta(i))/M} - 1\right)$
- N is number of segments in calculation
- M is MTBF for a 5000-node compute app (131572)
- F is matrix fill time (900)
- ρ is time to read the checkpoint sets (1440)
- T_S is total matrix factor time (81573)
- $\delta(i)$ is time to write checkpoint set i [δ(N)=0]
  - 960 * $\sqrt[3]{(N+1-i)/N}$

Selection of Checkpoint Count (2)

- Values of Tw for various choices of N

N	T_w	
1	116858	
2	99744	
3	95334	
4	93631	
5	92946	
6	92572	Optimal
7	92832	

Other Pliris-C/R Resilience Features

- **EIGER job script enhancements**
  - On aprun termination, checks stdout/stderr for signs of recoverable conditions (node failures) and relaunches within the job using spare node(s)
- **Pliris_cr**
  - Tool to set up, verify, and clean up checkpoint sets
  - Saves on file open times in parallel application
  - Helps with scratch directory hygiene
- **Pliris_watch**
  - Tool to watch running EIGER job, and report/act on signs of stalls

Results from EIGER Runs with Pliris-C/R

- **First successful run 4/24/2014 (Job 1474501)**
  - 6 checkpoint writes: 956 sec → 871 sec
  - 1 checkpoint read/restart: 1435 sec
  - Performance compares well with fs_test and other turnstiling apps
- **Strange run 11/25/2014 (Job 1568851)**
  - 7 checkpoint writes: 2826 sec → 2004 sec
  - 1 checkpoint read/restart: 2019 sec
  - Full file system? Overlapped with file system directory tree walk?
- **Latest run 2/27/2015 (Job 1627163)**
  - Assertion failed in MPI_Barrier: recv_pending (BUG 824088)

Areas of Future Work

- Port to Trinity (DataWarp + DNE)
- Skip matrix fill on restart run
- First-come, first-served queueing on turnstiles
- Improve checkpoint interval
  - Closer to optimal
  - Adjustable in restart runs
- Overlap I/O on static portion of matrix with factorization of active portion

Summary

- Adding C/R to a dense solver is viable
- Turnstiling still helps I/O
- Shared file descriptors can help I/O
- Good citizenship promotes resiliency

Acknowledgements

- **Courtenay T. Vaughn (SNL), Brett M. Kettering (LANL), Dan Poznanovic (CRAY)**
  - Reviewed paper and gave valuable feedback
- **William W. Tucker (formerly Cray Inc.)**
  - Coauthor
- **Joseph D. Kotulski (SNL)**
  - Coauthor

Q&A