A Framework for Check-Pointed Fault-Tolerant Out-of-Core Linear Algebra

Ed D’Azevedo (e6d@ornl.gov)
Oak Ridge National Laboratory
Piotr Luszczek (luszczech@cs.utk.edu)
University of Tennessee
Acknowledgement

• Research performed as part of Science Application Pilot Program (SAPP) in support of SciDAC project “Numerical Calculations of Wave-Plasma Interactions in Multi-dimensional Systems” within the Office of Fusion Energy Sciences.

• Funding support from U. S. Department of Energy under Contract No. DE-AC05-00OR22725 with UT-Battelle, LLC.
Out-of-core ScaLAPACK

• Out-of-core version of ScaLAPACK LU linear solver allows the solution of problems several times larger than available memory.

• Full matrix stored on disk. Factorization routine transfers submatrices into memory.

• The 'Left-looking' LU factorization approach is used with in-core block column panels.

• ScaLAPACK used for in-core computations.

• Much more efficient than just “paging”.
I/O

- ScaLAPACK uses 2D block cyclic decomposition to store distributed matrix.
- Same data to processor mapping used for out-of-core matrix.
- Record oriented I/O. Each record is a $\text{MMB}$ by $\text{NNB}$ submatrix in ScaLAPACK format, where record size is multiple of blocksize, $\text{mod} (\text{MMB}, \text{MB} \times \text{NPROW}) = 0 = \text{mod} (\text{NNB}, \text{NB} \times \text{NPCOL})$ on $\text{NPROW}$ by $\text{NPCOL}$ processor grid.
I/O

- Unaligned transfers are supported with message communications.
- Record size can be tuned for I/O subsystem (say RAID stripe in 64KBytes).
- Can use multiple files to exceed 2GBytes per file limit.
- Common option uses separate files per processor on local disk (/tmp on linux cluster).
Check-point/Restart Capability

- Runs may take many hours or even several days.
- Time limit in batch queue policy (12hr limit at Center for Computational Sciences, ORNL).
- Low MTBF, especially on Linux cluster built with off-the-shelf components.
- Factors and partial results still on disk. Conceptually simple to restart computation.
Approach

- Generate micro-instructions in file. These instructions are mapped to subroutine calls such as read in panel, write out panel, perform update, or in core factorization.
- Simple driver to process instructions. Driver write out partial results and last instruction location before stopping.
- Details: driver look ahead in instruction stream to find next write command in checkpoint; driver look back for read command to restore panel in memory.
Micro-instructions
Micro-instructions

- Instruction encodes Scalapack subroutine names, e.g. PDGEMM
- Complete list of parameters for subroutine call, including matrix descriptors, indices and offsets.
- Only several hundred instructions even for a large problem.
Writing in 2-Steps

- The factored panel is first written out to a temporary location. Out-of-core matrix is not affected if this fails. Repeated (duplicate) computation may be needed for that panel during restart.

- If this is successful, the out-of-core matrix is then updated. Even if a machine crash corrupts the matrix, the intact data in temporary location can be used for recovery.

- Details: other (empty) temporary files are used to mark progress, e.g. 00143a.dat (00143b.dat) indicate first (second) write is successful for instruction 143.
Limitations

• Need to restart with the same set of processors if a local file system (e.g. /tmp on linux cluster) is used.

• Assume data is automatically “flushed” to disk after the file is closed. Delayed writes may cause problem on machine crash since data is still cached in memory.
Performance

• The checkpoint/restart imposes very low overhead over original ScaLAPACK out-of-core solver.

• Linux cluster with single 2Ghz P4 with 768MBytes memory, 30GBytes available in /tmp, 100Mbits ethernet switch, LAM/MPI, ATLAS BLAS (nonSSE2).

• Real*8, N=56000, each processor dedicate 288MBytes to solver, MB=NB=50.
• On 2x2 processors, check-point version took 32210sec (8.95hr), non check-point version took 31274sec (8.69hr), or about 3% overhead.

• About 913Mflops/cpu in check-point version.

• In-core ScaLAPACK solution on 7x7 processors took 3012sec (0.84hr) or about 793Mflops/cpu.

• Read matrix took 29.8sec (18.4MBytes/cpu)

• Write matrix took 44.2sec (11.6MBytes/cpu)
Summary

- Check-point/restart enhancement of out-of-core ScaLAPACK linear solver.
- Micro-instruction file allows easy way to stop and resume computations.
- 2-Step writes to recover from machine crash.
- Low overhead over non check-point version.