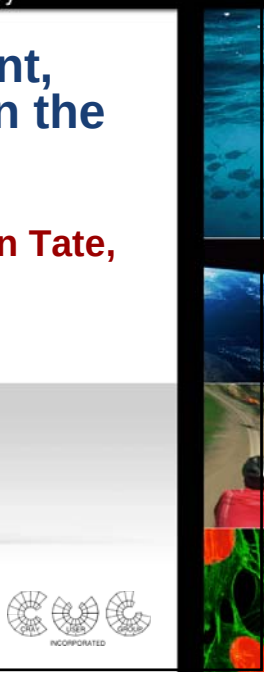
CRAY The Supercomputer Company

Performance Measurement, Tuning, and Optimization on the Cray XT3

Luiz DeRose, John Levesque, Adrian Tate, Cray Inc

- CUG 2006



Outline

- Single processor optimization
- Scientific Library performance tips
- Cray performance tools

CRAY The Supercomputer Company

Optimization for the Cray XT3™ MPP Supercomputer

- CUG 2006

John M. Levesque

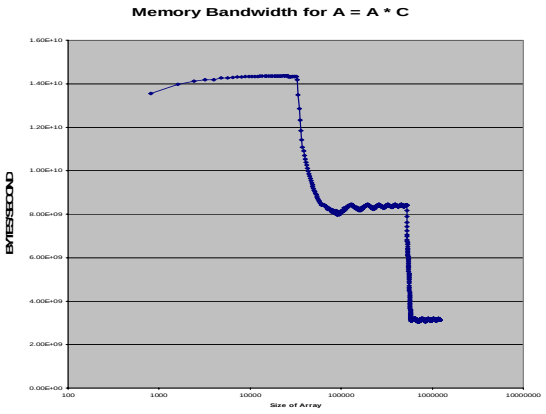
May, 2006

CRAY The Supercomputer Company

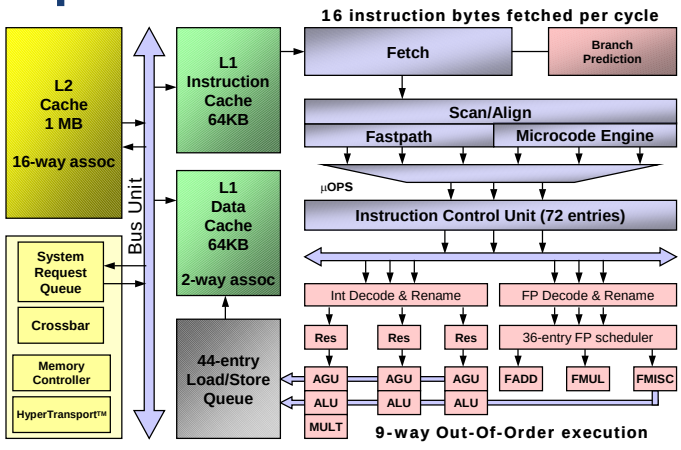
Outline of Optimization Section

- Single node (socket) Optimization
- I/O optimization

Bandwidth as a function of Array Size

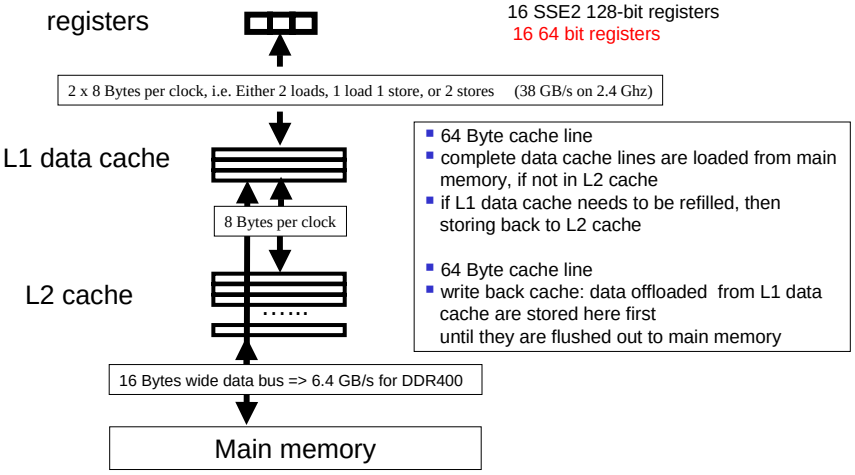


AMD Opteron Processor



- 36 entry FPU instruction scheduler
- 64-bit/80-bit FP Realized throughput (1 Mul + 1 Add)/cycle: 1.9 FLOPs/cycle
- 32-bit FP Realized throughput (2 Mul + 2 Add)/cycle: 3.4+ FLOPs/cycle

Simplified memory hierachy on the AMD Opteron



- 64 Byte cache line
- complete data cache lines are loaded from main memory, if not in L2 cache
- if L1 data cache needs to be refilled, then storing back to L2 cache

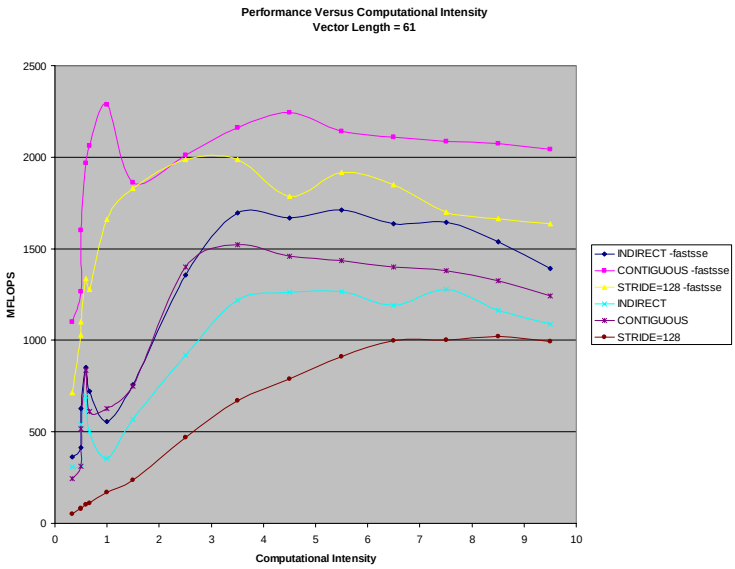
- 64 Byte cache line
- write back cache: data offloaded from L1 data cache are stored here first until they are flushed out to main memory

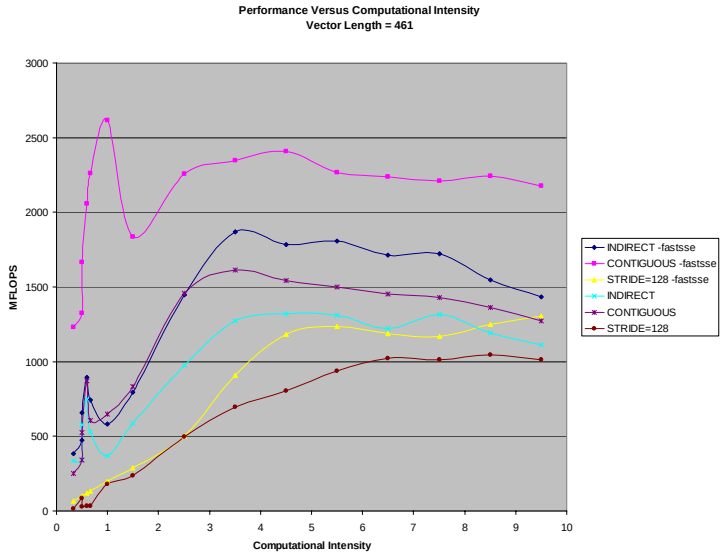

```
C     11 OPERATIONS - 2 OPERANDS     RATIO = 11/2
      II=1
      ISTRIDE = 128
      DO 41075 I = 1, N
       Y(II) = c0 + X(II) * (C1 + X(II) * (C2 + X(II)
     *                    * (C3 + X(II) * (C4 + X(II)
     *                    * (C5 + X(II)              )))))
       II = II + ISTRIDE
41075 CONTINUE
```

```
C     3 OPERATIONS - 5 OPERANDS     RATIO = 3/5

      DO 41023 I=1, N
       A(I) = B(I) * C(I) + D(I) * E(I)
41023 CONTINUE
```



```
C     17 OPERATIONS - 2 OPERANDS     RATIO = 17/2
      DO LLLLL =1,NREPS

      DO 41018 I = 1,N
       Y(IY(I)) =  c0 + X(IX(I)) * (C1 + X(IX(I))
     *          * (C2 + X(IX(I)) * (C3 + X(IX(I))
     *          * (C4 + X(IX(I)) * (C5 + X(IX(I))
     *          * (C6 + X(IX(I)) * (C7 + X(IX(I))
     *          * (C8 + X(IX(I))                 ))))))))
41018 CONTINUE
```

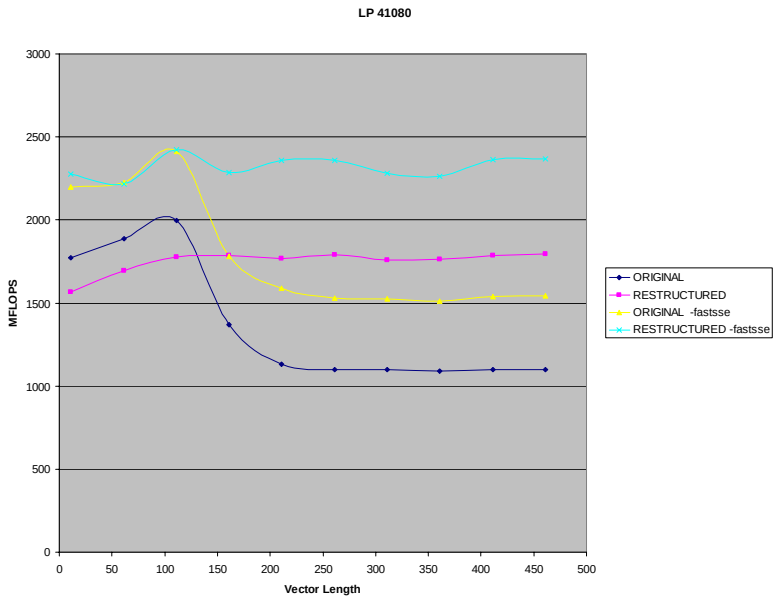

Performance Versus Computational Intensity
Vector Length = 61
MFLOPS
Computational Intensity
INDIRECT -fastsse
CONTIGUOUS -fastsse
STRIDE=128 -fastsse
INDIRECT
CONTIGUOUS
STRIDE=128


Performance Versus Computational Intensity
Vector Length = 461
MFLOPS
Computational Intensity
INDIRECT -fastsse
CONTIGUOUS -fastsse
STRIDE=128 -fastsse
INDIRECT
CONTIGUOUS
STRIDE=128

```
C     DIMENSION A(128,N)

      DO 41080  I = 1,N
       A( 1,I) = C1*A(13,I) + C2* A(12,I) + C3*A(11,I) +
     *           C4*A(10,I) + C5* A( 9,I) + C6*A( 8,I) +
     *           C7*A( 7,I) + C0*(A( 5,I) + A( 6,I) )  + A( 3,I)
41080 CONTINUE
```

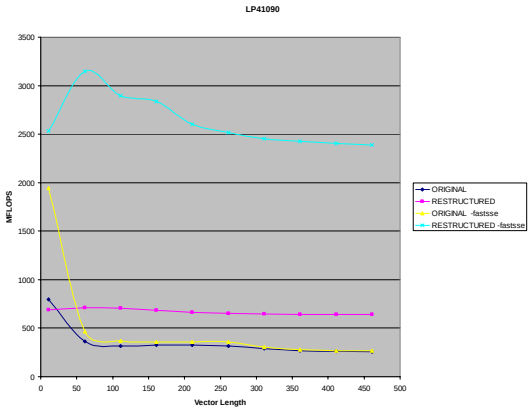


```
C     DIMENSION B(13,N)

      DO 41081 I = 1,N
       B( 1,I) = C1*B(13,I) + C2* B(12,I) + C3*B(11,I) +
     *           C4*B(10,I) + C5* B( 9,I) + C6*B( 8,I) +
     *           C7*B( 7,I) + C0*(B( 5,I) + B( 6,I) )  + B( 3,I)
41081 CONTINUE
```




```
C      THE ORIGINAL

      DO 41090 K = KA, KE, -1
        DO 41090 J = JA, JE
          DO 41090 I = IA, IE
            A(K,L,I,J) = A(K,L,I,J) - B(J,1,i,k)*A(K+1,L,I,1)
     *    - B(J,2,i,k)*A(K+1,L,I,2) - B(J,3,i,k)*A(K+1,L,I,3)
     *    - B(J,4,i,k)*A(K+1,L,I,4) - B(J,5,i,k)*A(K+1,L,I,5)
41090 CONTINUE
```

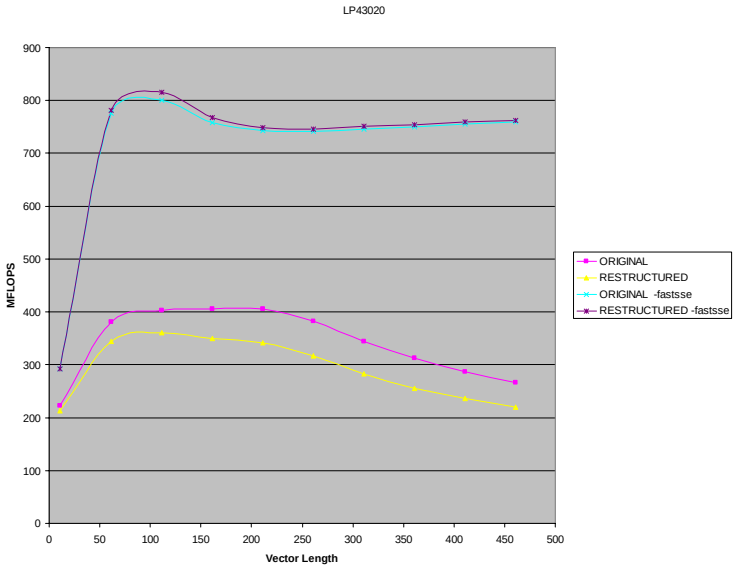
```
C      THE RESTRUCTURED

       DO 41091 K = KA, KE, -1
         DO 41091 J = JA, JE
           DO 41091 I = IA, IE
             AA(I,K,L,J) = AA(I,K,L,J)-BB(I,J,1,K)*AA(I,K+1,L,1)
     *     - BB(I,J,2,K)*AA(I,K+1,L,2)-BB(I,J,3,K)*AA(I,K+1,L,3)
     *     - BB(I,J,4,K)*AA(I,K+1,L,4)-BB(I,J,5,K)*AA(I,K+1,L,5)
41091 CONTINUE
```

```
      C  GAUSS ELIMINATION

      DO 43020 I = 1, MATDIM
       A(I,I) = 1. / A(I,I)
       DO 43020 J = I+1, MATDIM
        A(J,I) = A(J,I) * A(I,I)
        DO 43020 K = I+1, MATDIM
         A(J,K) = A(J,K) - A(J,I) * A(I,K)
43020 CONTINUE
```

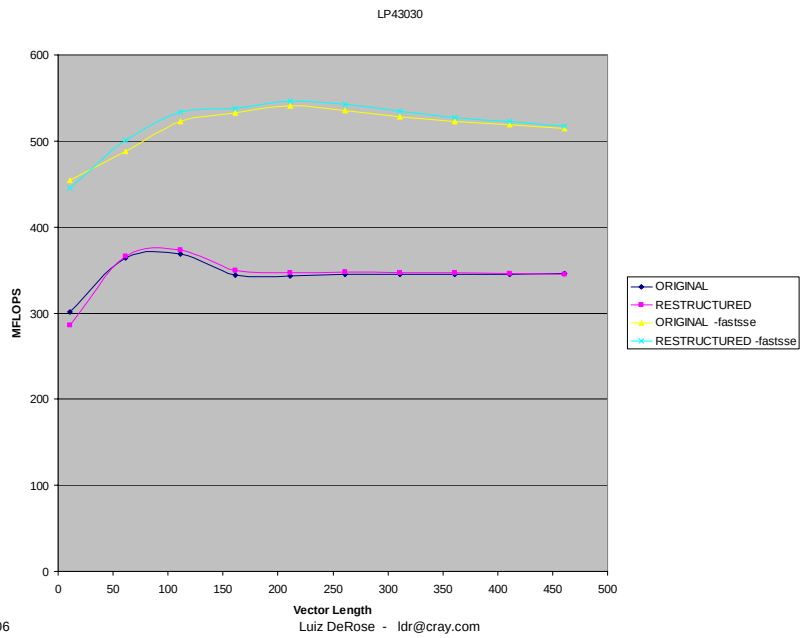


```
      C  GAUSS ELIMINATION

      DO 43020 I = 1, MATDIM
       A(I,I) = 1. / A(I,I)
       DO 43020 J = I+1, MATDIM
        A(J,I) = A(J,I) * A(I,I)
cpgi$l nodepchk
        DO 43020 K = I+1, MATDIM
         A(J,K) = A(J,K) - A(J,I) * A(I,K)
43020 CONTINUE
```




```
C      THE ORIGINAL

       DO  43030  I = 2, N
       DO  43030  K = 1, I-1
        A(I)= A(I)  + B(I,K) * A(I-K)
43030 CONTINUE
```

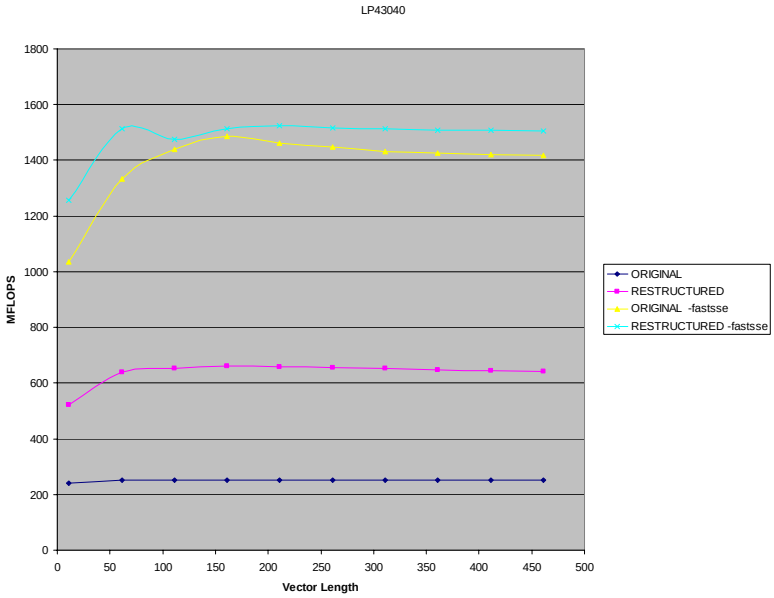
```
      DO 43031 I = 2, N
cpgi$l nodepchk
       DO 43031 K = 1, I-1
        A(I) = A(I)  + B(I,K) * A(I-K)
43031 CONTINUE
```

```
      DO 43040 J = 2, 8
      N1 = J
      N2 = J - 1
      DO 43040 I = 2, N
       A(I,N1) = A(I-1,N2) * B(I,J) + C(I)
43040 CONTINUE
```

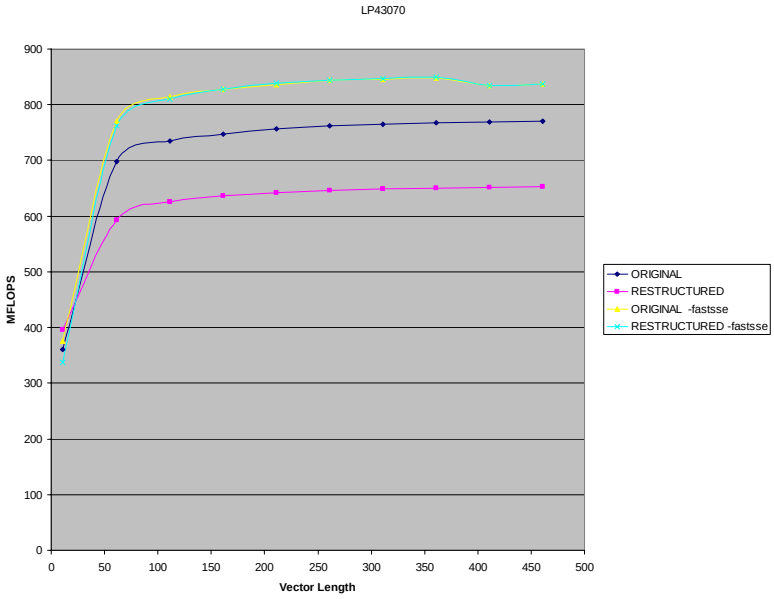


```
     DO 43041 J = 2, 8
      DO 43041 I = 2, N
       A(I,J) = A(I-1,J-1) * B(I,J) + C(I)
43041 CONTINUE
```




```
C      THE ORIGINAL

       DO 43070 I = 1, N
       A(IA(I)) = A(IA(I)) + C0 * B(I)
43070 CONTINUE
```

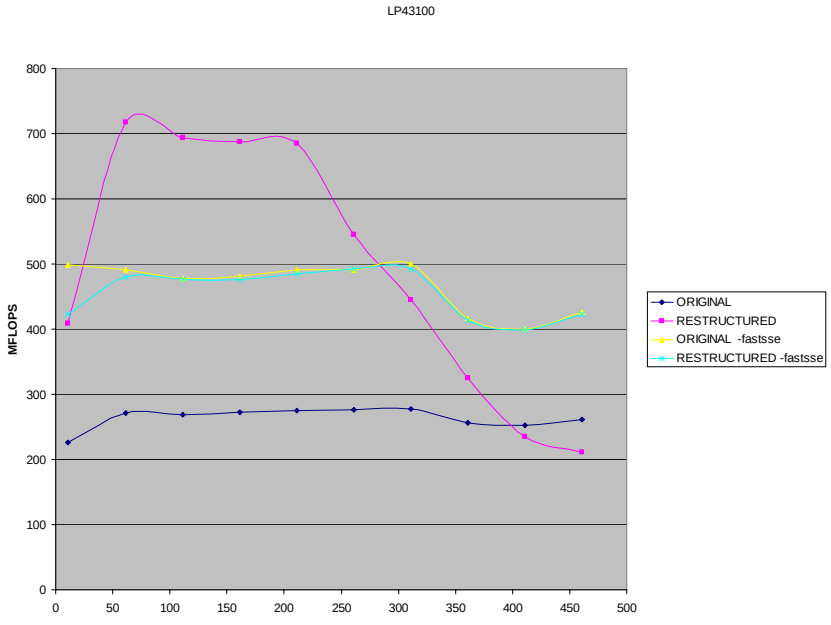
```
cpgi$l nodepchk
      DO 43071 I = 1, N
       A(IA(I)) = A(IA(I)) + C0 * B(I)
43071 CONTINUE
```

```
C      THE ORIGINAL
       DO 43100 J = 2, N
       AH = B(J) - B(J-1)
       DO 43100 I = 2, N
        A(I,J) = AH * A(I-1,J) + C(I,J)
43100 CONTINUE
```

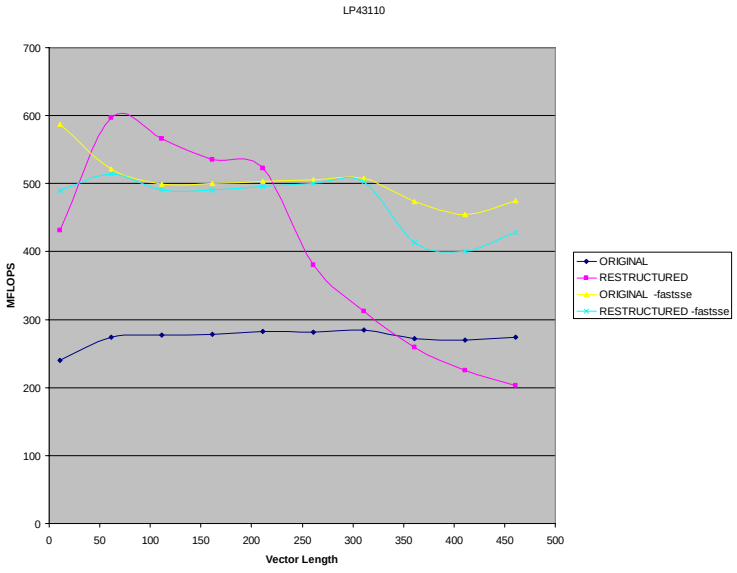


```
      DO 43101 J = 2, N
       VAH(J) = B(J) - B(J-1)
43101 CONTINUE

      DO 43102 I = 2, N
       DO 43102 J = 1, N
        A(I,J) = VAH(J) * A(I-1,J) + C(I,J)
43102 CONTINUE
```




```
C      THE ORIGINAL

      DO 43111 J = 2, N
       AH = B(J) - B(J-1)
       DO 43110 I = 2, N
        A(I,J) = AH  *  A(I-1,J) + C(I,J)
43110  CONTINUE

       BH  =  D(J) - D(J-1)
       DO 43112 I  =  N, 2,  - 1
        A(I,J)  =  BH  *  A(I+1,J) + C(I,J)
43112  CONTINUE

43111 CONTINUE
```

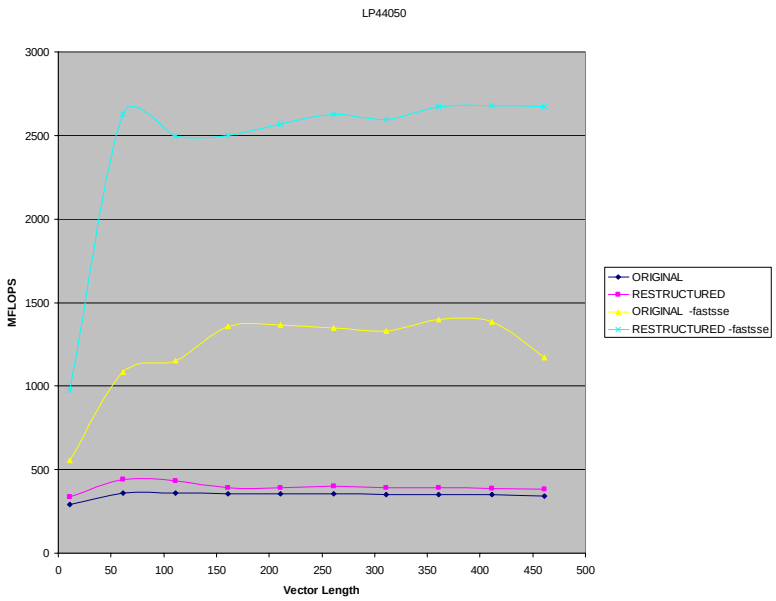
```
C      THE RESTRUCTURED

       DO 43113 J = 2, N
        VAH(J) = B(J) - B(J-1)
43113  CONTINUE

       DO 43114 I = 2, N
        DO 43114 J  =  2, N
         A(I,J) = VAH(J)  *  A(I-1,J) + C(I,J)
43114  CONTINUE

       DO 43115 J  =  2, N
        VBH(J)  =  D(J) - D(J-1)
43115  CONTINUE

       DO 43116 I  =  N, 2,  - 1
        DO 43116 J  =  2, N
         A(I,J)  =  VBH(J)  *  A(I+1,J) + C(I,J)
43116  CONTINUE
```

```
      DO 44050 I = 1, N
      DO 44050 J = 1, N
       A(I,J) = 0.0
       DO 44050 K = 1, N
        A(I,J) = A(I,J) + B(I,K) * C(K,J)
44050 CONTINUE
```

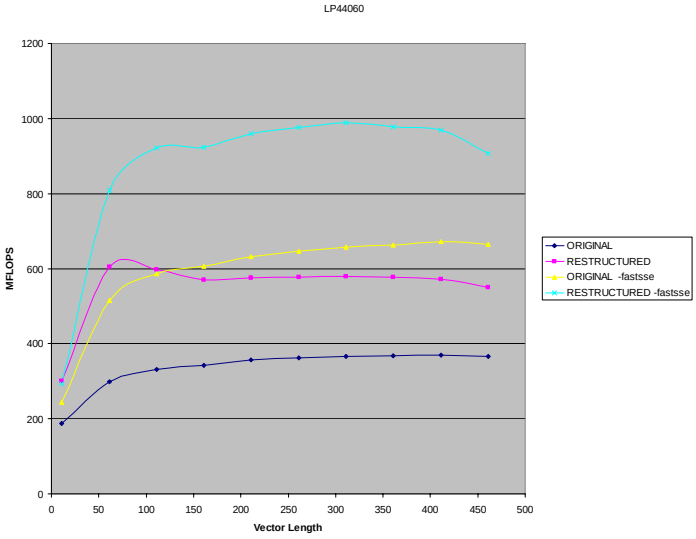


```
      DO 44051 J = 1, N
       DO 44051 I = 1, N
        A(I,J) = 0.0
44051 CONTINUE

      DO 44052 K = 1, N
       DO 44052 J = 1, N
        DO 44052 I = 1, N
         A(I,J) = A(I,J) + B(I,K) * C(K,J)
44052 CONTINUE
```




```
      DO 44060 I = 1, N
      A(I) = 0.0
      DO 44060 J = 1, I
       A(I) = A(I) + B(I,J) * C(J,I)
44060 CONTINUE
```

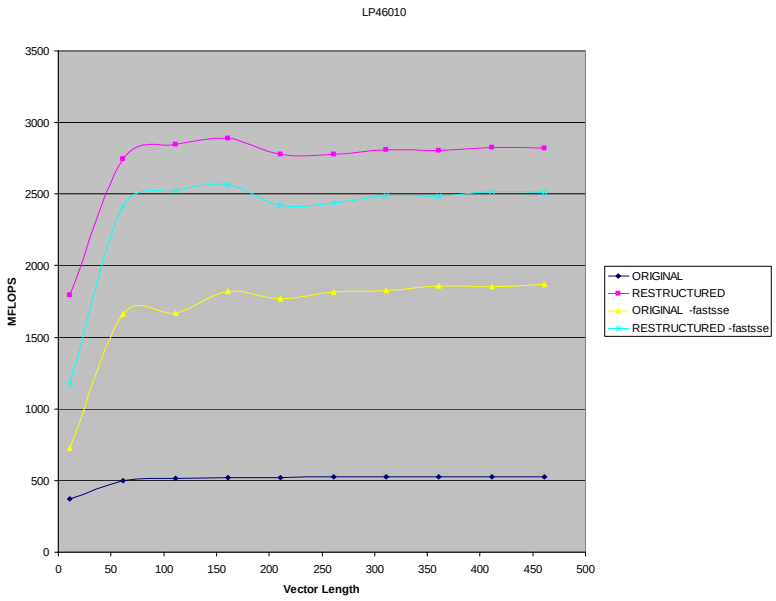
```
      DO 44061 I = 1, N
       A(I) = 0.0
44061 CONTINUE

      DO 44062 J = 1, N
       DO 44062 I = J, N
        A(I) = A(I) + B(I,J) * C(J,I)
44062 CONTINUE
```

```
C      THE ORIGINAL
      DO 46011 J = 1, 4
       DO 46010 I = 1, N
        C(J,I)=0.0
46010 CONTINUE
        DO 46011 K = 1,4
        DO 46011 I = 1,N
         C(J,I) = C(J,I) + A(J,K) * B(K,I)
46011 CONTINUE
```

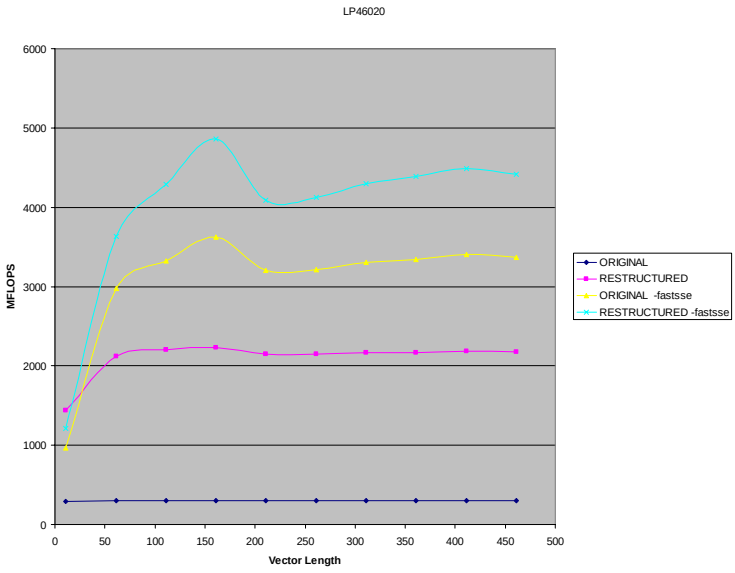


```
C       THE RESTRUCTURED
      DO 46012 I = 1, N
        C(1,I) = A(1,1) * B(1,I) + A(1,2) * B(2,I)
     *         + A(1,3) * B(3,I) + A(1,4) * B(4,I)
        C(2,I) = A(2,1) * B(1,I) + A(2,2) * B(2,I)
     *         + A(2,3) * B(3,I) + A(2,4) * B(4,I)
        C(3,I) = A(3,1) * B(1,I) + A(3,2) * B(2,I)
     *         + A(3,3) * B(3,I) + A(3,4) * B(4,I)
        C(4,I) = A(4,1) * B(1,I) + A(4,2) * B(2,I)
     *         + A(4,3) * B(3,I) + A(4,4) * B(4,I)
46012 CONTINUE
```




```
C      THE ORIGINAL

      DO 46020 I = 1,N
       DO 46020 J = 1,4
        A(I,J) = 0.
        DO 46020 K = 1,4
         A(I,J) = A(I,J) + B(I,K) * C(K,J)
46020 CONTINUE
```

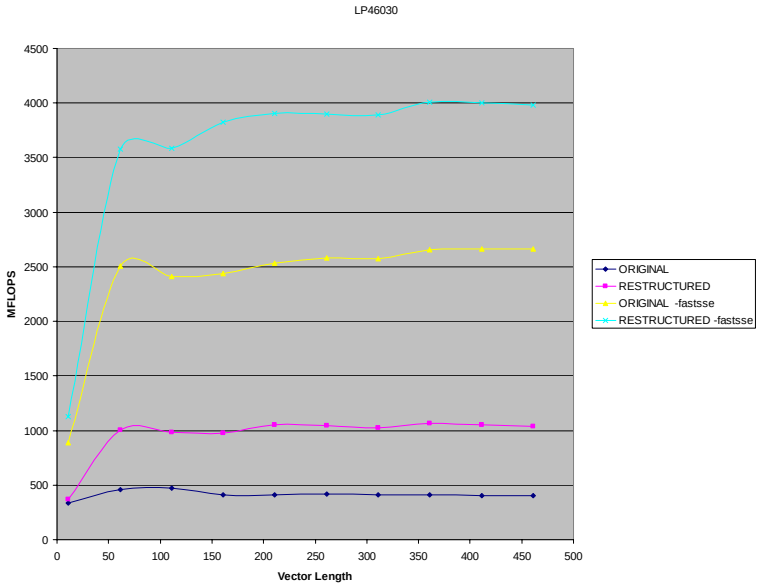
```
      DO 46021 I = 1, N
        A(I,1) = B(I,1) * C(1,1) + B(I,2) * C(2,1)
     *         + B(I,3) * C(3,1) + B(I,4) * C(4,1)
        A(I,2) = B(I,1) * C(1,2) + B(I,2) * C(2,2)
     *         + B(I,3) * C(3,2) + B(I,4) * C(4,2)
        A(I,3) = B(I,1) * C(1,3) + B(I,2) * C(2,3)
     *         + B(I,3) * C(3,3) + B(I,4) * C(4,3)
        A(I,4) = B(I,1) * C(1,4) + B(I,2) * C(2,4)
     *         + B(I,3) * C(3,4) + B(I,4) * C(4,4)
46021 CONTINUE
```

```
      DO 46030 J  = 1, N
       DO 46030 I = 1, N
        A(I,J) = 0.
46030 CONTINUE

      DO 46031   K = 1, N
       DO 46031  J = 1, N
        DO 46031 I = 1, N
         A(I,J) = A(I,J) + B(I,K) * C(K,J)
46031 CONTINUE
```



```
C       THE RESTRUCTURED

      DO 46032  J = 1, N
       DO 46032 I = 1, N
        A(I,J)=0.
46032 CONTINUE
C
      DO 46033   K = 1, N-5, 6
       DO 46033  J = 1, N
        DO 46033 I = 1, N
         A(I,J) = A(I,J) + B(I,K  ) * C(K  ,J)
     *                   + B(I,K+1) * C(K+1,J)
     *                   + B(I,K+2) * C(K+2,J)
     *                   + B(I,K+3) * C(K+3,J)
     *                   + B(I,K+4) * C(K+4,J)
     *                   + B(I,K+5) * C(K+5,J)
46033 CONTINUE
C
      DO 46034  KK = K, N
       DO 46034  J = 1, N
        DO 46034 I = 1, N
         A(I,J) = A(I,J) + B(I,KK) * C(KK ,J)
46034 CONTINUE
```



I/O Optimization

- Lustre
  - Stripe Setting
- I/O Buffering

Determining Stripe Size

- `lfs find -v /lustre/scratch/jlbeck`

```
OBDS:
0: ost1_UUID ACTIVE
...
15: ost16_UUID ACTIVE
/lustre/scratch/jlbeck/ has no stripe info
```

- Notice this directory has no stripe information. This means the width is ONE. If I were to create a file in this directory it would inherit the properties of the directory.

Stripe setting

- What should my stripe width be?
  - Lots of processors writing to individual files: sw = 1
  - 1 I/O process writing for all, sw = small > 1
  - Lots of processes all writing to 1 file. Sw = large
- Set your stripe width
  - lfs setstripe <filename> <stripe-size> <start-ost> <strip count>
    - lfs setstripe file 0 –1 5
    - lfs setstrip file 0 0 -1

IOBUF

- **iobuf** is an I/O buffering library. **iobuf** intercepts the I/O calls (open, read, etc.) from a program and provides an additional layer of buffering. In the case of XT3, iobuf replaces the stdio (glibc/libio) layer of buffering. By asynchronously prefetching and caching file data, I/O wait time for programs which read or write large file sequentially can be reduced
- **iobuf** can gather run time statistics and print a summary report of I/O activity for each file.
- If a memory allocation error occurs, buffering is reduced or disabled for that file and a diagnostic is printed to stderr. When the file is opened, a single buffer is allocated if buffering is enabled. The allocation of additional buffers is done when a buffer is needed. When a file is closed, its buffers are freed unless asynchronous I/O is still pending on the buffer

IOBUF

- File selection and parameters for buffering is set through the IOBUF_PARAMS environment variable. The default if IOBUF_PARAMS is not set is no buffering (the I/O call is passed onto the next layer without intervention). The general format is a comma-separated list of specifications.
  - IOBUF_PARAMS=spec1,spec2,spec3,...

Pattern	Description
*	Matches any number of characters
?	Matches any single character
[a-b]	Matches single character between **a** and **b**, inclusive
\	Interprets the following meta-character literally (quotes it)

IOBUF

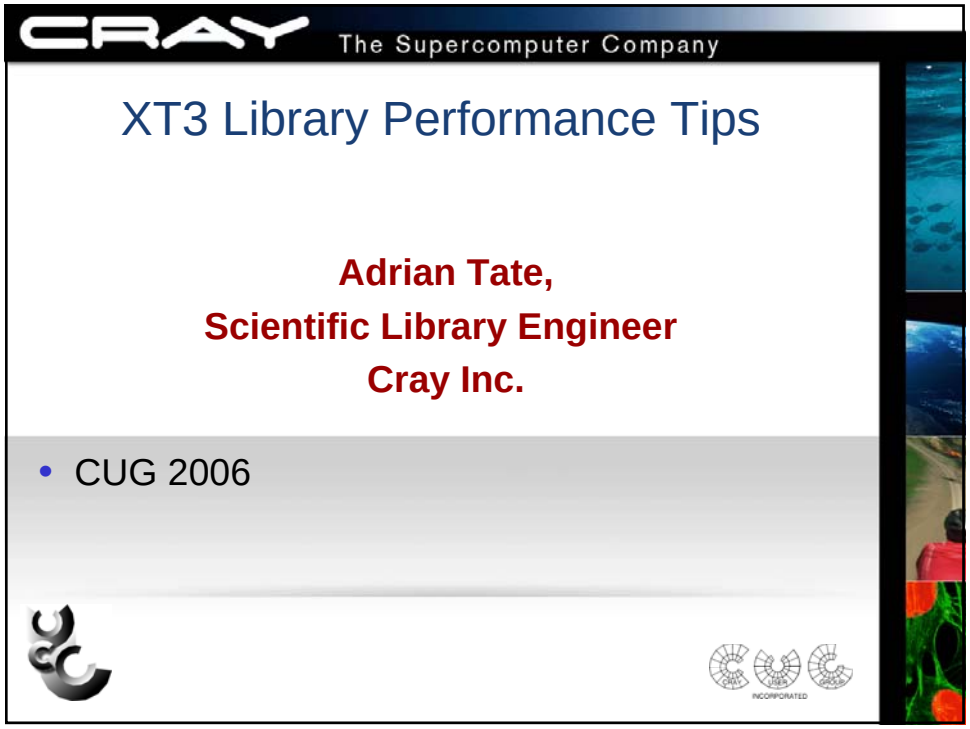
%stdin	Set the parameters for standard input.
%stdout	Set the parameters for standard output.
%stderr	Set the parameters for standard error.
%shared	Set the parameters for the shared cache. The default is 16 buffers of size 1M.

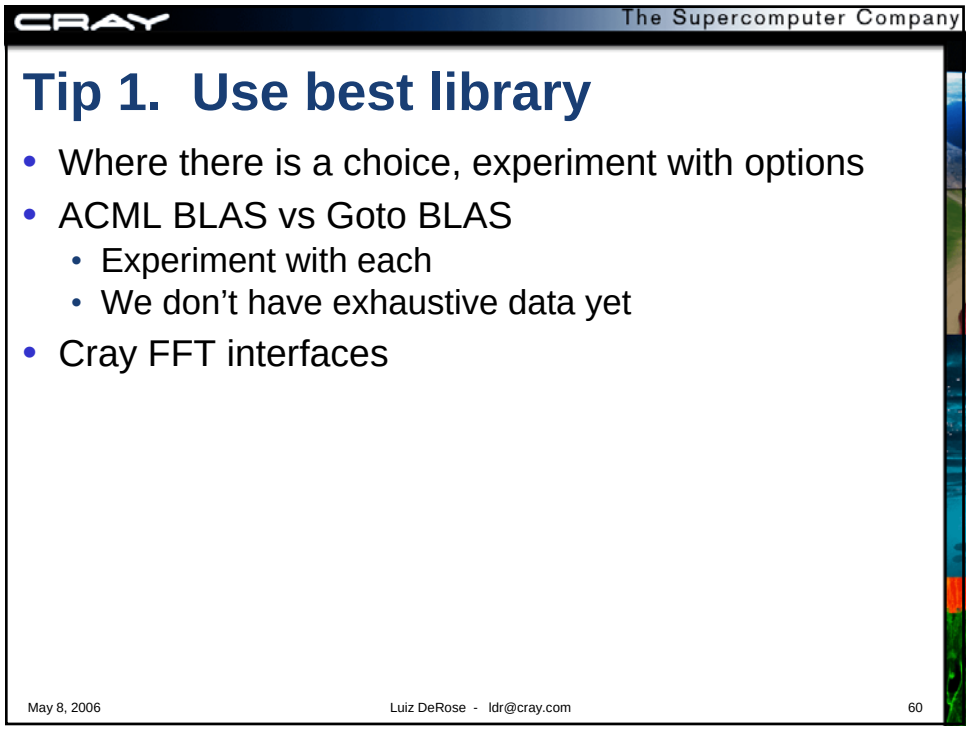
Parameter	Description
Count = n	The number of buffers. The default is 4.
Size = n	The size of the buffers. A suffix of **K**, **M**, or **G** can be given for kilobytes, megabytes, or gigabytes, respectively. The default is 1M.
prefetch=**n**	The number of buffers which are asynchronously prefetched ahead of the current buffer in use. The default is 1, meaning that the next buffer is automatically prefetched. Setting this value to 0 will disable prefetching.

Example

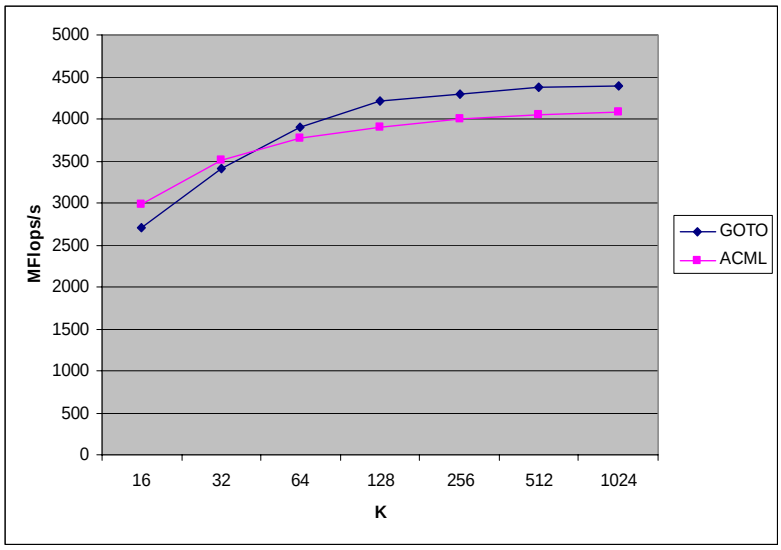
export IOBUF_SIZE=32768
export IOBUF_COUNT=5
export IOBUF_PARAMS='*/*/*:eagerflush:lazyclose,%stdout:eagerflush:lazyclose'

Parameter	Description
eagerflush	Flush buffers as soon as they are full. If this is not set, the buffers are flushed only when no buffers are available or when the file is closed. This option is useful when streaming data out to a file because it allows for some parallel overlap between producing the data and writing it.
lazyclose	If lazyclose is set, I/O requests (fetching or flushing) which have been started but are not complete are left outstanding when the file is closed. If this is not set, then all asynchronous requests are completed before the file is closed. This is useful if the timing of the written data is not critical.


CRAY
The Supercomputer Company
XT3 Library Performance Tips
Adrian Tate,
Scientific Library Engineer
Cray Inc.
• CUG 2006


CRAY
The Supercomputer Company
Tip 1. Use best library
• Where there is a choice, experiment with options
• ACML BLAS vs Goto BLAS
• Experiment with each
• We don't have exhaustive data yet
• Cray FFT interfaces
May 8, 2006
Luiz DeRose - ldr@cray.com
60

ACML vs GOTO zgemm M=1024 N=1024



Tip 2. 32-bit is faster

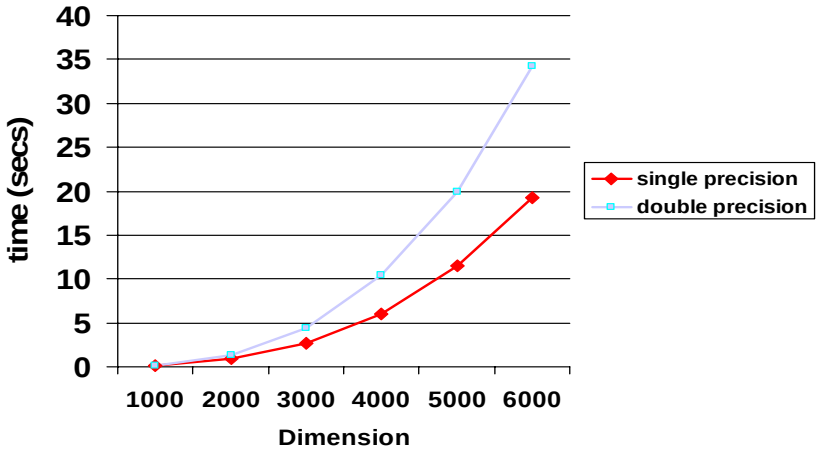
- 32-bit arithmetic on Opteron can be ~40% quicker
  - sse instructions of length 4
- Can use Residual Correction (iterative refinement) to retain precision if you have some memory to spare

e.g. LU with refinement

- Ax=b
- LUx=b with single precision but keep a copy of A in double precision
  - Generate r = b – Ax in double precision
  - Solve to find new x_1 using single precision solver
  - $x = x + x_1$
  - Iterate
- If matrix has a high condition number, refinement will take too long

Double vs Single + refinement

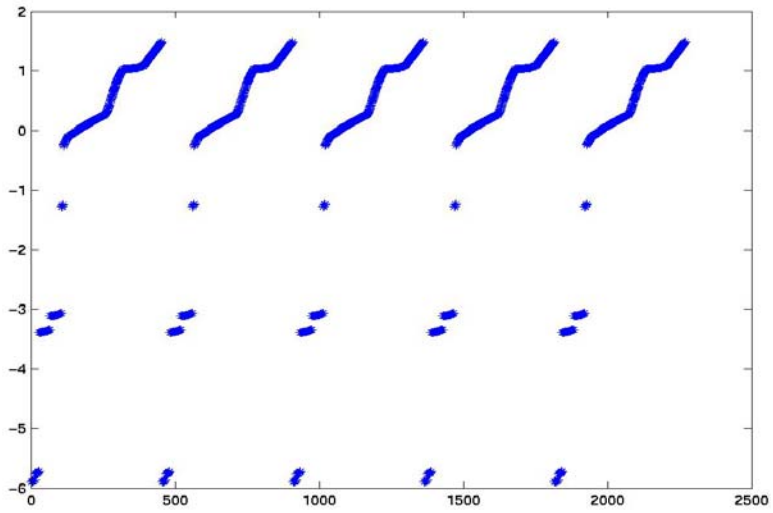
-



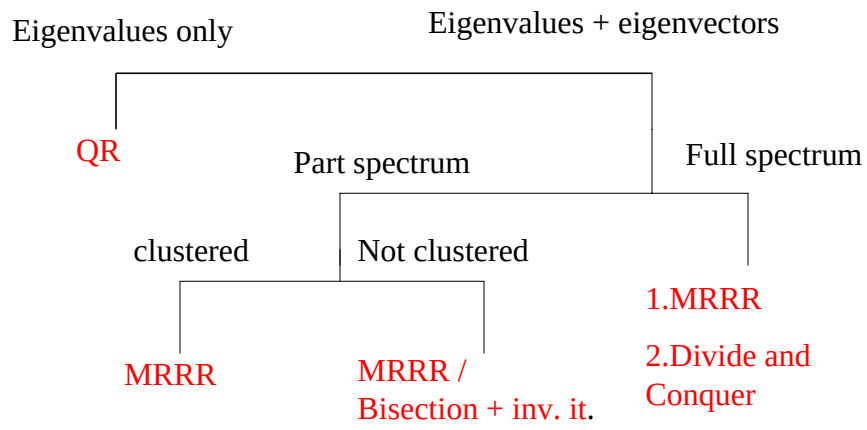
Tip 3. Know your data

- Solver performance can depend on characteristics of your matrix
- Eigensolvers will be wildly different for various data types.
- Generate a graphic of your eigenvalue spectrum

e.g. clustered eigenvalues



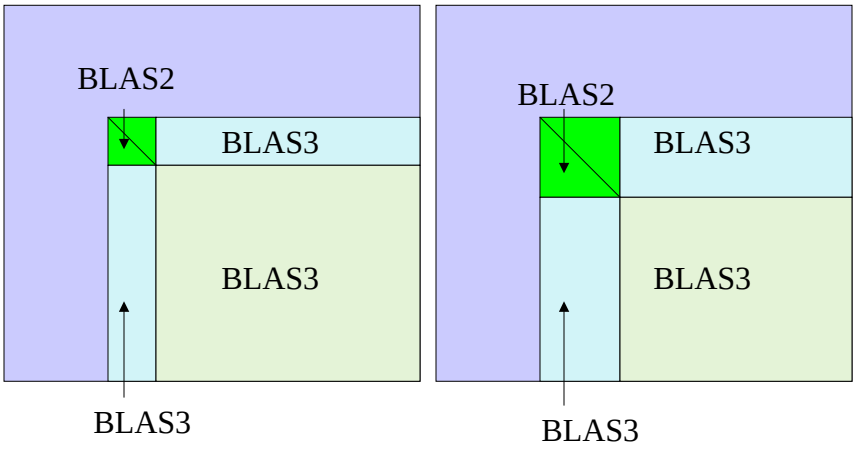
Decision tree for eigensolvers (XT3)



Tip 4. Know your parameters

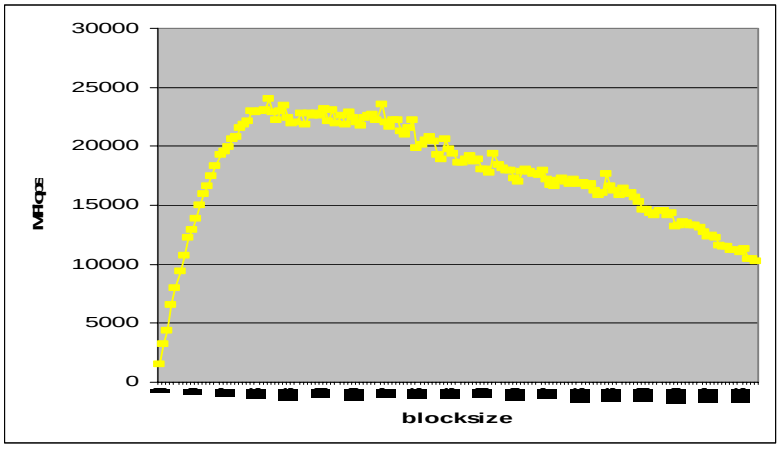
- For parallel - increase blocksize
  - Many codes use same on different machines
- XT3 MPI implementation works best for fewer, larger messages
  - bigger distribution block size best
  - Up to a point

1 stage of LU - limit of block size growth



Blocksize variance for Cholesky

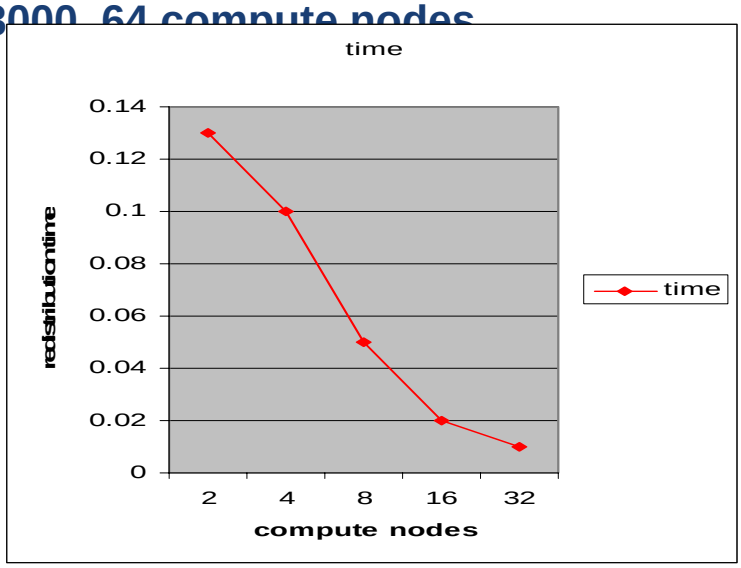
-



Paramters cont

- Redistribute into optimal blocksize between code portions
- Redistribute is cheap compared to advantages

Redistribution is cheap
M=3000, 64 compute nodes



Parameters cont

- Use optimal decomposition
  - ScaLAPACK grid shape
  - FFT butterflies

Tip 5. Go sparse

- Nearly all real data is sparse
- Sparse solvers are $O(n^2)$
- Cray XT-libsci will include iterative solvers in future release
- AMD will support direct solvers in future release

Summary

- Use ACML and GOTO BLAS when they are fastest
- Try using 32-bit with iterative refinement
- Choose correct algorithm for your data's character
- Increase blocksize as far as you can
- Redistribute within your code
- Use optimal decomposition
- Go Sparse

Cray Performance Analysis Tools

Luiz DeRose
Programming Environment Director
Cray Inc.

- CUG 2006

The Cray Tools Strategy

- Must be easy to use
  - Automatic program instrumentation
    - no source code or makefile modification needed
- Integrated performance tools solution
- Strategy based on the three main steps normally used for application optimization and tuning:
  - Debug application
  - Single processor optimization
  - Parallel processing and I/O optimization
- Close interaction with user for feedback targeting functionality enhancements

Cray Performance Analysis Infrastructure

- **CrayPat**
  - pat_hwpc: for whole program measurement
  - pat_build: Utility for application instrumentation
    - No source code modification required
  - run-time library for measurements
    - transparent to the user
  - pat_report:
    - Performance reports
    - Performance visualization file
  - libhwpc
  - pat_help
- **Cray Apprentice^2**
  - Graphical performance analysis and visualization tool
  - Can be used off-line on Linux system

CrayPat API

- CrayPat performs automatic instrumentation at function level
- The CrayPat API can be used for fine grain instrumentation
  - Fortran
    - call PAT_region_begin(id, “label”, ierr)
    - DO Work
    - call PAT_region_end(id, ierr)
  - C
    - include <pat_api.h>
    - …
    - ierr = PAT_region_begin(id, “label”);
    - DO_Work();
    - ierr = PAT_region_end(id);

Performance Data Collection

- Two dimensions
  - When Performance Collection is triggered
    - External agent (asynchronous)
      - Sampling (not yet supported on the XT3)
        - » timer interrupt
        - » hardware counters overflow
    - Internal agent (synchronous)
      - Code instrumentation (event trace)
        - » Automatic instrumentation
        - » Hand instrumentation
  - How performance data is recorded
    - Profile (runtime summary)
    - Trace file

Six Steps for Performance Analysis

1. Load CrayPat module
2. Build application
   - No makefile modification needed
3. Instrument application with pat_build
   - % pat_build [-g group] [-u] [options] a.out
     - Groups: mpi, io, heap, user function (-u) …
     - Automatic instrumentation at group (function) level
       - No source code modification needed
4. Run instrumented application
5. Generate performance file (.ap2) with pat_report
   - % pat_report –f ap2 [options] <directory with .xf files>
6. Performance analysis and visualization with CrayPat and Cray Apprentice^2

Runtime Environment Variables

- The following runtime environment variables affect how the data is collected:
  - PAT_RT_REGION_MAX
    - Specifies the largest numerical ID that may be used as an argument to the CrayPat API functions PAT_region_begin and PAT_region_end
      - The default is 100
  - PAT_RT_SUMMARY
    - Enables run-time summarization
      - Includes the aggregation of data during run-time
      - Runtime summarization is enabled by default
  - PAT_RT_HWPC <set #>
    - Activate collection of hardware performance counters
      - There are 9 sets on the XT3

pat_report Options

- pat_report [-V] [-b b-opts] [-d d-opts] [-f ap2|txt|xml] [-i dir|instrprog] [-O keyword|options_file] [-o output_file] [-s key=value] [-T] [-t cum_threshold] data_file|directory ...
  - Arguments can be any .xf, .xml, .ap2 suffixed data_file produced by CrayPat, or a directory containing them
  - Default is to produce a summary report for the performance data in the specified data_file or directory

- Main options:
  - -i is only if the instrumented program has a different name or is in a different directory path than when it was executed
  - **-O specifies the type of report**:
    - profile, ca (callers), ca+src, ct (calltree) ct+src, lb (load_balance), lb_all (load_balance_all), mpi
  - -b, -d, -s can be used to further customize the report

Pat_report Output

All profiles have this information, which is helpful to link the data with a particular execution

```
CrayPat/X:  Version 3.0 Revision 131 (xf 73)  04/12/06 08:58:30

Experiment:  trace

Experiment data file:
  /lus/nid00008/ldr/sweep3d/sweep3d+pat+25/sweep3d+pat+25tdo-*.xf  (RTS)

Original program:  /lus/nid00008/ldr/sweep3d/sweep3d

Instrumented program:  /lus/nid00008/ldr/sweep3d/sweep3d+pat

Program invocation:  sweep3d+pat

Number of PEs:  24

Runtime environment variables:  PAT_RT_HWPC=1

Report time environment variables:
  PAT_ROOT=/opt/xt-tools/craypat/3.0/cpatx

Report command line options:  <none>

Host name and type:  guppy x86_64  2400 MHz

Operating system:  catamount 1.0 2.0

Hardware performance counter events:
  PAPI_TLB_DM  Data translation lookaside buffer misses
  PAPI_L1_DCA  Level 1 data cache accesses
  PAPI_FP_OPS  Floating point operations
  DC_MISS      Data Cache Miss
  User_Cycles  Virtual Cycles

Traced functions:
  MAIN_                    .../nid00008/ldr/sweep3d/driver.f
  MPI_Abort                ==NA==
  MPI_Allreduce            ==NA==
  MPI_Attr_put             ==NA==
. . .
```

Table 1: Flat Profile (Default)

```
Table 1:  -d time%@0.05,cum_time%,time,traces,P
          -b exp,group,function,pe=...

  This table shows only lines with Time% > 0.05.
  Percentages at each level are relative
    (for absolute percentages, specify:  -s percent=a).

  Time% | Cum.Time% |     Time |  Calls |Experiment=1
                                        |Group
                                         |Function
                                          |PE='HIDE'

 100.0% |    100.0% | 4.158385 | 483556 |Total
|-----------------------------------------------------
|  65.4% |     65.4% | 2.720561 | 245332 |USER
||----------------------------------------------------
||  96.8% |     96.8% | 2.634367 |    576 |sweep_
||   1.7% |     98.5% | 0.046572 |    576 |source_
||   0.3% |     98.9% | 0.008985 |    576 |flux_err_
||   0.3% |     99.2% | 0.007825 |     48 |inner_
||   0.3% |     99.4% | 0.007594 | 118080 |snd_real_
||   0.3% |     99.7% | 0.007492 |     48 |MAIN_
||   0.2% |     99.9% | 0.004699 | 118080 |rcv_real_
||====================================================
|  34.6% |    100.0% | 1.437824 | 238224 |MPI
||----------------------------------------------------
||  58.7% |     58.7% | 0.843810 | 118080 |mpi_recv_
||  20.9% |     79.5% | 0.299967 |    144 |mpi_barrier_
||   9.8% |     89.4% | 0.140919 |   1536 |mpi_allreduce_
||   6.6% |     96.0% | 0.095254 |    192 |mpi_bcast_
||   4.0% |    100.0% | 0.057867 | 118080 |mpi_send_
|====================================================
```

New feature: The report will only show functions with at least 0.05% of the time

Table 2: MPI Profile Total (Default)

```
Table 2:  -d time%@0.05,time,sc,sm,sz
          -b exp,group,pe=[mmm]

  This table shows only lines with Time% > 0.05.
  Percentages at each level are relative
    (for absolute percentages, specify:  -s percent=a).

  Time% |     Time | SentMsgCt | SentMsgTot | SentMsgSz |Experiment=1
                                                        |Group
                                                         |PE[mmm]

 100.0% | 4.158385 |    118080 | 1244160000 |  10536.59 |Total
|-------------------------------------------------------------------
|  65.4% | 2.720561 |        -- |         -- |        -- |USER
||------------------------------------------------------------------
||   2.6% | 3.438231 |        -- |         -- |        -- |pe.0
||   2.1% | 2.734409 |        -- |         -- |        -- |pe.8
||   2.0% | 2.612655 |        -- |         -- |        -- |pe.43
||==================================================================
|  34.6% | 1.437824 |    118080 | 1244160000 |  10536.59 |MPI
||------------------------------------------------------------------
||   2.3% | 1.563542 |      2160 |   21081600 |   9760.00 |pe.44
||   2.1% | 1.440358 |      2160 |   23846400 |  11040.00 |pe.12
||   1.2% | 0.804185 |      1440 |   15206400 |  10560.00 |pe.0
|==================================================================
```

New feature:
Send msg profile with call path & load balance information

Table 3: MPI Profile (Default)

```
Table 3:  -d time%@0.05,time,sc@,sm,sz
          -b exp,function,callers,pe=[mmm]

  This table shows only lines with:
    Time% > 0.05
    SentMsgCt > 0
  Percentages at each level are relative
    (for absolute percentages, specify:  -s percent=a).

  Time% |     Time | SentMsgCt | SentMsgTot | SentMsgSz |Function
                                                         |Caller
                                                         | PE[mmm]
  100.0% | 4.158385 |    118080 | 1244160000 |  10536.59 |Total
|-------------------------------------------------------------------------
|  34.6% | 1.437824 |    118080 | 1244160000 |  10536.59 |MPI
||------------------------------------------------------------------------
||   4.0% | 0.057867 |    118080 | 1244160000 |  10536.59 |mpi_send_
||        |          |           |            |           | snd_real_
||        |          |           |            |           |  sweep_
||        |          |           |            |           |   inner_
||        |          |           |            |           |    inner_auto_
||        |          |           |            |           |     MAIN_
|||||||||-----------------------------------------------------------------
|||||||||   2.6% | 0.071167 |      2880 |   30412800 |  10560.00 |pe.26
|||||||||   2.0% | 0.055372 |      2160 |   23846400 |  11040.00 |pe.29
|||||||||   1.3% | 0.036089 |      1440 |   15206400 |  10560.00 |pe.5
|=========================================================================
```

New feature:
Send msg profile with call path & load balance information

Call Tree Profile (Top Down)

```
% pat_build -O ct <performance file>

Time% | Cum.Time% |        Time |     Calls |Calltree

 100.0% |    100.0% | 2647.397216 | 283222164 |Total
|-----------------------------------------------------
| 100.0% |    100.0% | 2647.396498 | 283221076 |main
||----------------------------------------------------
||  99.9% |     99.9% | 2644.447759 | 283220884 |MAIN_
|||---------------------------------------------------
|||  98.0% |     98.0% | 2594.643913 | 283218250 |runhyd_
||||--------------------------------------------------
||||  15.6% |     15.6% |  412.532275 |  47186240 |yzsweep_
|||||-------------------------------------------------
|||||  13.6% |     13.6% |  359.658178 |  47185920 |sppm2_
|||||-------------------------------------------------
||||||   6.8% |      6.8% |  179.125866 |   5242880 |sppm2_(exclusive)
||||||   3.4% |     10.1% |   89.007423 |   5242880 |difuze_
||||||   2.5% |     12.6% |   65.459283 |  26214400 |interf_
||||||   1.0% |     13.6% |   26.065606 |  10485760 |dintrf_
|||||=================================================
|||||   2.0% |     15.6% |   52.874097 |       320 |yzsweep_(exclusive)
|||||=================================================
||||  15.5% |     31.1% |  411.367894 |  47186240 |zysweep_
||||-------------------------------------------------
|||||  13.5% |     29.1% |  358.397289 |  47185920 |sppm2_
|||||-------------------------------------------------
||||||   6.7% |     22.3% |  178.614169 |   5242880 |sppm2_(exclusive)
||||||   3.4% |     25.7% |   89.178333 |   5242880 |difuze_
||||||   2.4% |     28.1% |   64.800752 |  26214400 |interf_
||||||   1.0% |     29.1% |   25.804036 |  10485760 |dintrf_
|||||=================================================
```

Callers Profile (Bottom Up)

```
% pat_report -O ca <performance file>
 Time% | Cum.Time% |        Time |     Calls |Function
                                             |Caller

100.0% |    100.0% | 2647.397216 | 283222164 |Total
|-----------------------------------------------------------
|  40.5% |     40.5% | 1071.748077 |  31457280 |sppm2_
||----------------------------------------------------------
||   6.8% |      6.8% |  179.125866 |   5242880 |yzsweep_
||                                               | runhyd_
||                                               |  MAIN_
||                                               |   main
||   6.8% |     13.5% |  178.901135 |   5242880 |zzsweep_
||                                               | runhyd_
||                                               |  MAIN_
||                                               |   main
||   6.8% |     20.3% |  178.709675 |   5242880 |yxsweep_
||                                               | runhyd_
||                                               |  MAIN_
||                                               |   main
||   6.7% |     27.0% |  178.614169 |   5242880 |zysweep_
||                                               | runhyd_
||                                               |  MAIN_
||                                               |   main
||   6.7% |     33.8% |  178.273669 |   5242880 |xxsweep_
||                                               | runhyd_
||                                               |  MAIN_
||                                               |   main
||   6.7% |     40.5% |  178.123564 |   5242880 |xysweep_
||                                               | runhyd_
||                                               |  MAIN_
||                                               |   main
|===========================================================
|  20.2% |     60.7% |  534.482687 |  31457280 |difuze_
||                                               | sppm2_
||----------------------------------------------------------
|||  3.4% |     43.9% |   89.361514 |   5242880 |zzsweep_
|||                                               | runhyd_
|||                                               |  MAIN_
|||                                               |   main
```


Callers Profile with Line Numbers

```
% pat_report -O ca+src <performance file>
   Time% | Cum.Time% |        Time |     Calls |Function
                                             |Caller

 100.0% |    100.0% | 2647.397216 | 283222164 |Total
 |-----------------------------------------------------------
 |  40.5% |     40.5% | 1071.748077 |  31457280 |sppm2_
 ||----------------------------------------------------------
 ||   6.8% |      6.8% |  179.125866 |   5242880 |yzsweep_:/scratch/derose/sppm2/sweeps.F:line.518
 ||                                               | runhyd_:/scratch/derose/sppm2/main.F:line.1056
 ||                                               |  MAIN_:/scratch/derose/sppm2/main.F:line.226
 ||                                               |   main:NA:line.0
 ||   6.8% |     13.5% |  178.901135 |   5242880 |zzsweep_:/scratch/derose/sppm2/sweeps.F:line.812
 ||                                               | runhyd_:/scratch/derose/sppm2/main.F:line.1064
 ||                                               |  MAIN_:/scratch/derose/sppm2/main.F:line.226
 ||                                               |   main:NA:line.0
 ||   6.8% |     20.3% |  178.709675 |   5242880 |yxsweep_:/scratch/derose/sppm2/sweeps.F:line.1400
 ||                                               | runhyd_:/scratch/derose/sppm2/main.F:line.1080
 ||                                               |  MAIN_:/scratch/derose/sppm2/main.F:line.226
 ||                                               |   main:NA:line.0
 ||   6.7% |     27.0% |  178.614169 |   5242880 |zysweep_:/scratch/derose/sppm2/sweeps.F:line.1106
 ||                                               | runhyd_:/scratch/derose/sppm2/main.F:line.1072
 ||                                               |  MAIN_:/scratch/derose/sppm2/main.F:line.226
 ||                                               |   main:NA:line.0
 ||   6.7% |     33.8% |  178.273669 |   5242880 |xxsweep_:/scratch/derose/sppm2/sweeps.F:line.1694
 ||                                               | runhyd_:/scratch/derose/sppm2/main.F:line.1088
 ||                                               |  MAIN_:/scratch/derose/sppm2/main.F:line.226
 ||                                               |   main:NA:line.0
 ||   6.7% |     40.5% |  178.123564 |   5242880 |xysweep_:/scratch/derose/sppm2/sweeps.F:line.219
 ||                                               | runhyd_:/scratch/derose/sppm2/main.F:line.1048
 ||                                               |  MAIN_:/scratch/derose/sppm2/main.F:line.226
 ||                                               |   main:NA:line.0
 |===========================================================
 |  20.2% |     60.7% |  534.482687 |  31457280 |difuze_
 ||                                               | sppm2_:/scratch/derose/sppm2/sppm.F:line.630
 ||----------------------------------------------------------
 |||  3.4% |     43.9% |   89.361514 |   5242880 |zzsweep_:/scratch/derose/sppm2/sweeps.F:line.812
 |||                                               | runhyd_:/scratch/derose/sppm2/main.F:line.1064
 |||                                               |  MAIN_:/scratch/derose/sppm2/main.F:line.226
 |||                                               |   main:NA:line.0
 |||  3.4% |     47.2% |   89.178333 |   5242880 |zysweep_:/scratch/derose/sppm2/sweeps.F:line.1106
```

Load Balance: Max, Median, Min

```
% pat_report -O lb <performance file>

    Time% | Cum.Time% |     Time |   Calls |Experiment=1
                                          |Function
                                          |PE[mmm]
   100.0% |    100.0% | 2.447386 | 1022404 |Total
|-----------------------------------------------------
|  60.3% |     60.3% | 1.475444 |    1152 |sweep_
||----------------------------------------------------
||   1.1% |      1.1% | 1.566552 |      12 |pe.32
||   1.0% |     52.7% | 1.451626 |      12 |pe.88
||   0.9% |    100.0% | 1.299376 |      12 |pe.94
|=====================================================
|  31.0% |     91.3% | 0.759457 |  247680 |MPI_Recv
||----------------------------------------------------
||   2.4% |      2.4% | 1.745154 |    1440 |pe.95
||   1.0% |     58.3% | 0.738907 |    2160 |pe.23
||   0.6% |    100.0% | 0.471425 |    1440 |pe.0
|=====================================================
|   4.5% |     95.9% | 0.111178 |    3072 |MPI_Allreduce
||----------------------------------------------------
||   2.2% |      2.2% | 0.235157 |      32 |pe.0
||   1.0% |     70.8% | 0.108114 |      32 |pe.51
||   0.0% |    100.0% | 0.004355 |      32 |pe.95
|=====================================================
|   1.3% |     97.1% | 0.031291 |     288 |MPI_Barrier
||----------------------------------------------------
||   1.4% |      1.4% | 0.041278 |       3 |pe.94
||   1.0% |     52.2% | 0.031295 |       3 |pe.69
|| ...
```


Load Balancing Function per PE

```
%pat_report -O lb_all <performance file>

    100.0% |    100.0% | 2.447386 | 1022404 |Total
|-----------------------------------------------------
|  60.3% |     60.3% | 1.475444 |    1152 |sweep_
||----------------------------------------------------
||   1.1% |      1.1% | 1.566552 |      12 |pe.32
||   1.1% |      2.2% | 1.564838 |      12 |pe.45
|| ...
||   0.9% |     99.1% | 1.301373 |      12 |pe.95
||   0.9% |    100.0% | 1.299376 |      12 |pe.94
||====================================================
|  31.0% |     91.3% | 0.759457 |  247680 |MPI_Recv
||----------------------------------------------------
||   2.4% |      2.4% | 1.745154 |    1440 |pe.95
||   2.4% |      4.8% | 1.721688 |    2160 |pe.94
|| ...
||   0.8% |     99.4% | 0.578925 |    2160 |pe.8
||   0.6% |    100.0% | 0.471425 |    1440 |pe.0
||====================================================
|   4.5% |     95.9% | 0.111178 |    3072 |MPI_Allreduce
||----------------------------------------------------
||   2.2% |      2.2% | 0.235157 |      32 |pe.0
||   2.1% |      4.3% | 0.221896 |      32 |pe.8
|| ...
```

Load Balancing Flat Profile

```
pat_report –d ti%@0.05,ti,max_ti,min_ti <performance file>
  Table 1:  -d ti%@0.05,ti,max_ti,min_ti
          -b exp,function,pe=HIDE
  This table shows only lines with Time% > 0.05.
  Percentages at each level are relative
    (for absolute percentages, specify:  -s percent=a).

  Time% |     Time | Max.Time | Min.Time |Experiment=1
                                          |Function
 100.0% | 2.447386 | 2.447386 | 2.447386 |Total
|-----------------------------------------------------
|  60.3% | 1.475444 | 1.566552 | 1.299376 |sweep_
|  31.0% | 0.759457 | 1.745154 | 0.471425 |MPI_Recv
|   4.5% | 0.111178 | 0.235157 | 0.004355 |MPI_Allreduce
|   1.3% | 0.031291 | 0.041278 | 0.000212 |MPI_Barrier
|   1.3% | 0.031025 | 0.035856 | 0.016957 |MPI_Send
|   0.8% | 0.020509 | 0.025446 | 0.019012 |source_
|   0.2% | 0.004169 | 0.004553 | 0.003856 |flux_err_
|   0.1% | 0.003472 | 0.004011 | 0.001995 |snd_real_
|   0.1% | 0.003362 | 0.003983 | 0.001940 |rcv_real_
|   0.1% | 0.002867 | 0.003817 | 0.000060 |MPI_Bcast
|   0.1% | 0.002122 | 0.161605 | 0.000414 |inner_
|=====================================================
```


Hardware Performance Counters

- AMD Opteron Hardware Performance Counters
  - Four 48-bit performance counters.
    - Each counter can monitor a single event
      - Count specific processor events
        - » the processor increments the counter when it detects an occurrence of the event
        - » (e.g., cache misses)
      - Duration of events
        - » the processor counts the number of processor clocks it takes to complete an event
        - » (e.g., the number of clocks it takes to return data from memory after a cache miss)
  - Time Stamp Counters (TSC)
    - Cycles (user time)

Hardware Counters Selection

- PAT_RT_HWPC <set number> | <event list>
  - Specifies hardware counter events to be monitored
    - A set number can be used to select a group of predefined hardware counters events (recommended)
      - **CrayPat provides 9 sets on the Cray XT3**
    - Alternatively a list of hardware performance counter event names can be used
      - Maximum of 4 events
    - Both formats can be specified at the same time, with later definitions overriding previous definitions
    - By default, no hardware performance counter events are monitored during tracing experiments

Hardware Performance Counters

```
  PAPI_TLB_DM  Data translation lookaside buffer misses
  PAPI_L1_DCA  Level 1 data cache accesses
  PAPI_FP_OPS  Floating point operations
  DC_MISS      Data Cache Miss
  User_Cycles  Virtual Cycles
==========================================================
USER / sweep_
----------------------------------------------------------
  Time%                                        96.7%
  Cum.Time%                                    96.7%
  Time                                     2.634400
  Calls                                         576
  PAPI_TLB_DM               0.817M/sec       2154360 misses
  PAPI_L1_DCA           50663.437M/sec  133586218101 ops
  PAPI_FP_OPS           31741.840M/sec   83694919945 ops
  DC_MISS                1685.024M/sec    4442965950 ops
  User time                 2.637 secs    6328171597 cycles
  Utilization rate                             100.0%
  HW FP Ops / Cycles                            13.23 ops/cycle
  HW FP Ops / User time  31741.840M/sec   83694919945 ops        13.8%peak
  HW FP Ops / WCT        31741.840M/sec
  Computation intensity                          0.63 ops/ref
  LD & ST per TLB miss                       62007.38 ops/miss
  LD & ST per D1 miss                           30.07 ops/miss
  D1 cache hit ratio                            96.7%
  % TLB misses / cycle                           0.0%
```

PAT_RT_HWPC=1
Flat profile data
Hard counts
Derived metrics

PAT_RT_HWPC=2 (Cache Info)

```
  PAPI_L1_DCA          Level 1 data cache accesses
  DC_L2_REFILL_MOESI   Refill from L2. Cache bits: Modified Owner Exclusive Shared Invalid
  DC_SYS_REFILL_MOESI  Refill from system. Cache bits: Modified Owner Exclusive Shared Invalid
  BU_L2_REQ_DC         Internal L2 request - DC fill
  User_Cycles          Virtual Cycles
==========================================================================
sweep_
--------------------------------------------------------------------------
  Time%                                          49.2%
  Cum.Time%                                      49.2%
  Time                                       1.616597
  Calls                                          1152
  PAPI_L1_DCA              80105.593M/sec   129523883351 ops
  DC_L2_REFILL_MOESI        2115.140M/sec     3419999989 ops
  DC_SYS_REFILL_MOESI        558.142M/sec      902468278 ops
  BU_L2_REQ_DC              2186.443M/sec     3535290685 req
  User time                    1.617 secs     3880594435 cycles
  Utilization rate                               100.0%
  L1 Data cache misses      2673.282M/sec     4322468267 misses
  LD & ST per D1 miss                             29.97 ops/miss
  D1 cache hit ratio                              96.7%
  LD & ST per D2 miss                            143.52 ops/miss
  D2 cache hit ratio                              74.5%
  L2 cache hit ratio                              79.1%
  Memory to D1 refill        558.142M/sec      902468278 lines
  Memory to D1 bandwidth   34066.302MB/sec   57757969792 bytes
  L2 to Dcache bandwidth  129097.892MB/sec  218879999296 bytes
```


PAT_RT_HWPC=3 (L1 & L2 BW)

```
  PAPI_L1_DCM        Level 1 data cache misses
  PAPI_L1_DCA        Level 1 data cache accesses
  DC_L2_REFILL_MOES  Refill from L2. Cache bits: Modified Owner Exclusive Shared
  DC_COPYBACK_MOES   Copyback. Cache bits: Modified Owner Exclusive Shared
  User_Cycles        Virtual Cycles
==========================================================================
sweep_
--------------------------------------------------------------------------
  Time%                                          48.6%
  Cum.Time%                                      48.6%
  Time                                       1.616684
  Calls                                          1152
  PAPI_L1_DCM               2114.961M/sec     3420021828 misses
  PAPI_L1_DCA              80107.053M/sec   129538007948 ops
  DC_L2_REFILL_MOES         1556.843M/sec     2517510677 ops
  DC_COPYBACK_MOES          2671.538M/sec     4320040502 ops
  User time                    1.617 secs     3880946902 cycles
  Utilization rate                               100.0%
  LD & ST per D1 miss                             37.88 ops/miss
  D1 cache hit ratio                              97.4%
  Memory to D1 refill         558.118M/sec      902511151 lines
  Memory to D1 bandwidth    34064.826MB/sec   57760713664 bytes
  L2 to Dcache bandwidth    95022.165MB/sec  161120683328 bytes
  Dcache to L2 bandwidth   163057.741MB/sec  276482592128 bytes
```

PAT_RT_HWPC=4 (FP Mix)

```
  PAPI_FML_INS  Floating point multiply instructions
  PAPI_FAD_INS  Floating point add instructions
  PAPI_FP_OPS   Floating point operations
  FP_FAST_FLAG  Dispatched FPU ops that use the fast flag interface
  User_Cycles   Virtual Cycles
=============================================================================
sweep_
-----------------------------------------------------------------------------
  Time%                                        49.2%
  Cum.Time%                                    49.2%
  Time                                     1.616579
  Calls                                        1152
  PAPI_FML_INS           25428.085M/sec  41114651677 instr
  PAPI_FAD_INS           29646.485M/sec  47935378771 instr
  PAPI_FP_OPS            55074.569M/sec  89050030448 ops
  FP_FAST_FLAG            1514.677M/sec   2449080340 ops
  User time                  1.617 secs   3880558215 cycles
  Utilization rate                           100.0%
  HW FP Ops / Cycles                          22.95 ops/cycles
  HW FP Ops / User time  55074.569M/sec  89050030448 ops
  HW FP Ops / WCT        55085.464M/sec
  FP Multiply / FP Ops                        46.2%
  FP Add / FP Ops                             53.8%
```


PAT_RT_HWPC=6 (Stalls / Resources Idle)

```
  PAPI_FPU_IDL    Cycles floating point units are idle
  PAPI_STL_ICY    Cycles with no instruction issue
  PAPI_RES_STL    Cycles stalled on any resource
  IC_FETCH_STALL  Instruction fetch stall
  User_Cycles     Virtual Cycles
=============================================================================
sweep_
-----------------------------------------------------------------------------
  Time%                                        49.2%
  Cum.Time%                                    49.2%
  Time                                     1.616599
  Calls                                        1152
  PAPI_FPU_IDL           0.161 secs    387205898 cycles
  PAPI_STL_ICY           0.059 secs    141834156 cycles
  PAPI_RES_STL           0.966 secs   2319366199 cycles
  IC_FETCH_STALL         1.119 secs   2685785414 cycles
  User time              1.617 secs   3880617226 cycles
  Utilization rate                           100.0%
  Total time stalled     0.966 secs   2319366199 cycles  59.8%
  Time I Fetch Stalled   1.119 secs   2685785414 cycles  69.2%
  Avg Time FPUs idle     0.081 secs    193602949 cycles   5.0%
  Time Decoder empty     0.059 secs    141834156 cycles   3.7%
```

PAT_RT_HWPC=7 (Stalls/ Resources Full)

```
  FR_DECODER_EMPTY             Nothing to dispatch - decoder empty
  FR_DISPATCH_STALLS           Dispatch stalls - D2h or DAh combined
  FR_DISPATCH_STALLS_FULL_FPU  Dispatch stall when FPU is full
  FR_DISPATCH_STALLS_FULL_LS   Dispatch stall when LS is full
  User_Cycles                  Virtual Cycles
==========================================================================
sweep_
--------------------------------------------------------------------------
  Time%                                          49.1%
  Cum.Time%                                      49.1%
  Time                                       1.616695
  Calls                                          1152
  FR_DECODER_EMPTY             8421.217M/sec  13617375408 ops
  FR_DISPATCH_STALLS              0.967 secs   2319634916 cycles
  FR_DISPATCH_STALLS_FULL_FPU     0.399 secs    956897507 cycles
  FR_DISPATCH_STALLS_FULL_LS      0.077 secs    184231710 cycles
  User time                       1.617 secs   3880876277 cycles
  Utilization rate                               100.0%
  Total time stalled              0.967 secs   2319634916 cycles  59.8%
  Avg Time FPUs stalled           0.199 secs  478448753.5 cycles  12.3%
  Avg Time LSs stalled            0.038 secs     92115855 cycles   2.4%
  Time Decoder empty              5.674 secs  13617375408 cycles  350.9%
```


PAT_RT_HWPC Other Sets

```
Set 5: FP Mix (2)
  FR_FPU_X87              Retired FPU instructions - x87 instructions
  FR_FPU_MMX_3D           Retired FPU instructions - Combined MMX and 3DNow! instructions
  FR_FPU_SSE_SSE2_PACKED  Retired FPU instructions - Combined packed SSE and SSE2 Instr.
  FR_FPU_SSE_SSE2_SCALAR  Retired FPU instructions - Combined scalar SSE and SSE2 Instr.
  User_Cycles             Virtual Cycles


--------------------------------------------------------------------------
Set 8: Branches
  PAPI_BR_TKN   Conditional branch instructions taken
  PAPI_BR_MSP   Conditional branch instructions mispredicted
  PAPI_TOT_INS  Instructions completed
  IC_MISS       IC Miss
  User_Cycles   Virtual Cycles


--------------------------------------------------------------------------

Set 9: Instructions
  PAPI_L2_ICM   Level 2 instruction cache misses
  PAPI_L1_ICA   Level 1 instruction cache accesses
  IC_MISS       IC Miss
  IC_L2_REFILL  Refill from L2
  User_Cycles   Virtual Cycles
```

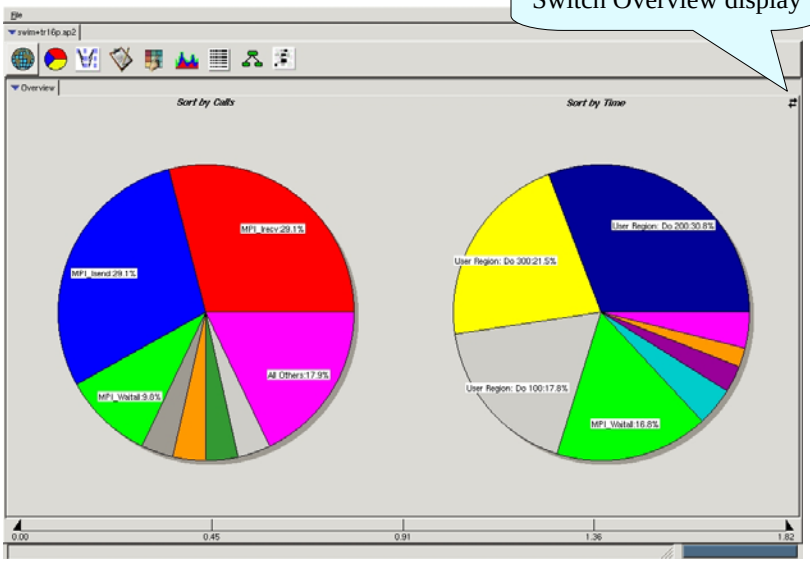
Cray Apprentice^2

- Call graph profile
- Communication statistics
- Time-line view
  - Communication
  - I/O
- Activity view
- Pair-wise communication statistics
- Text reports
- Source code mapping

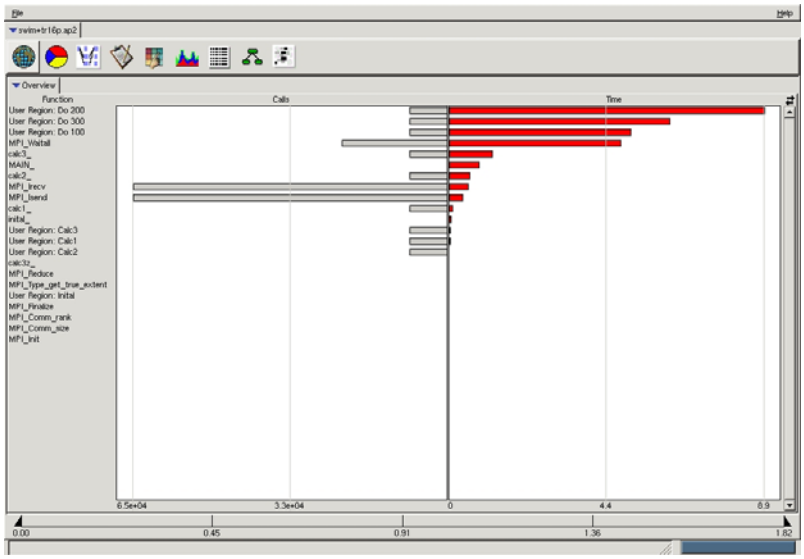
- Apprentice^2 is target to help identify and correct:
  - Excessive communication
  - Network contention
  - Load imbalance
  - Excessive serialization

Statistics Overview

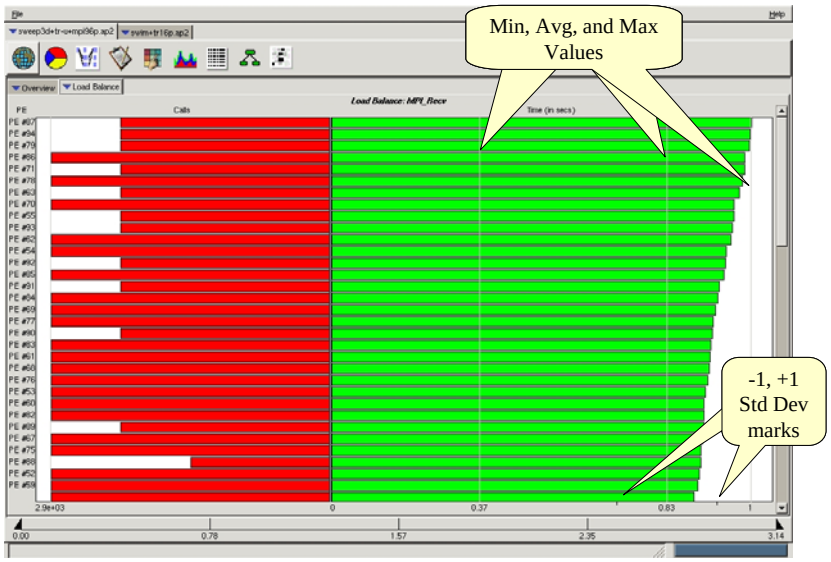
New feature:
Switch Overview display



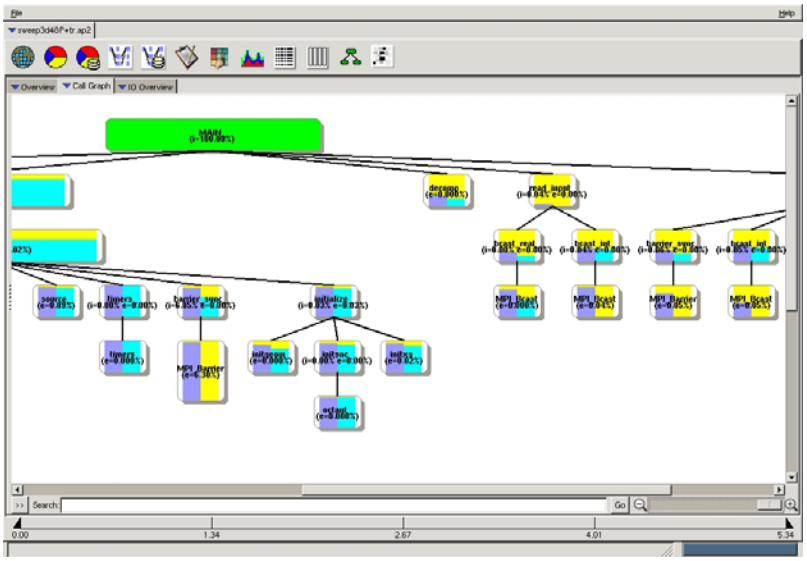
Function Profile



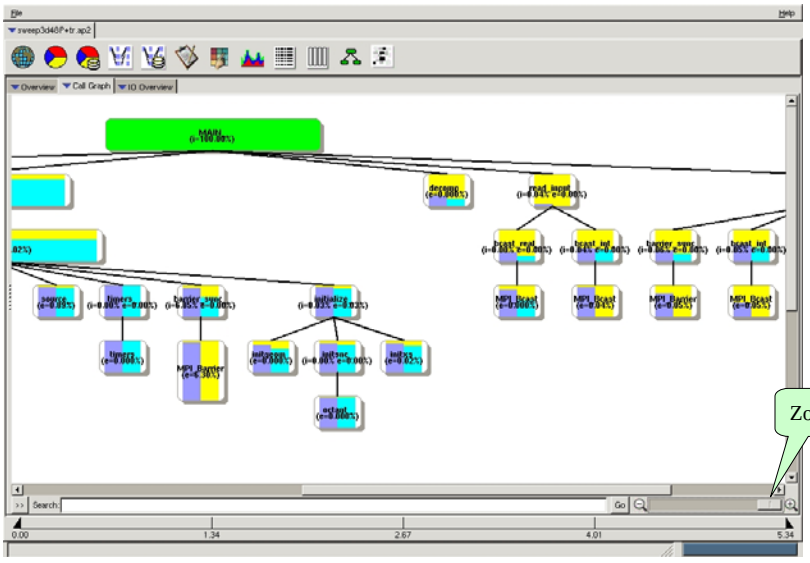
Load Balance View (Aggregated)



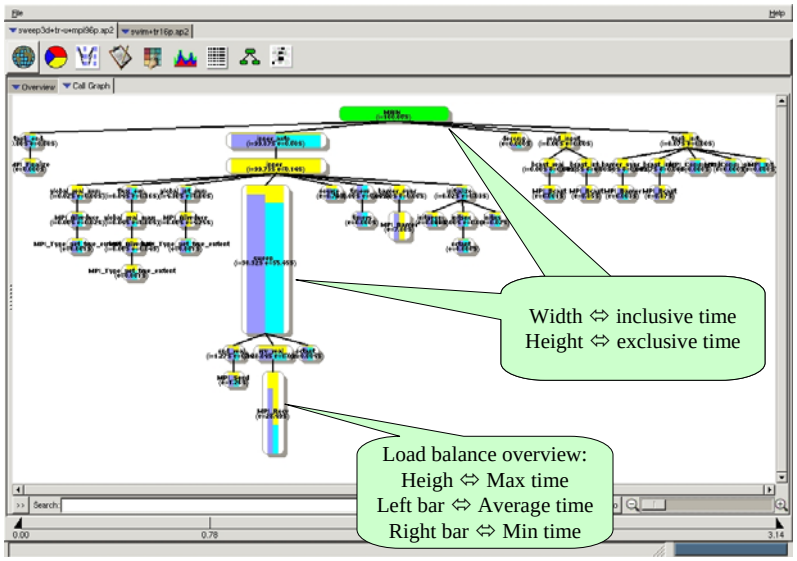
Call Graph View



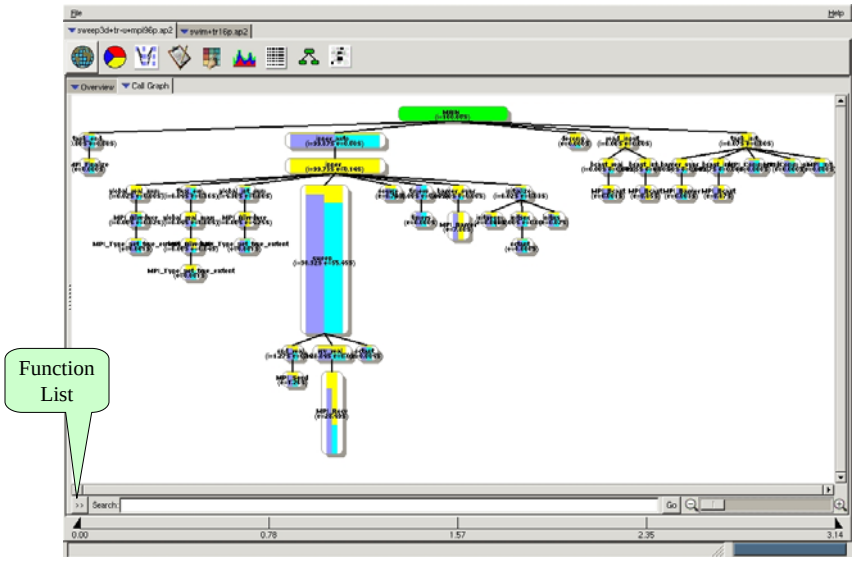
Call Graph View



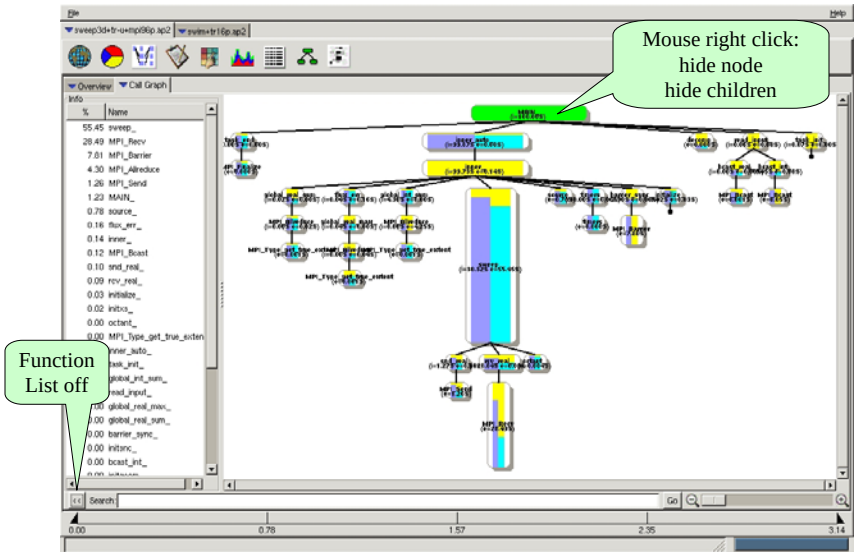
Call Graph View - Zoom



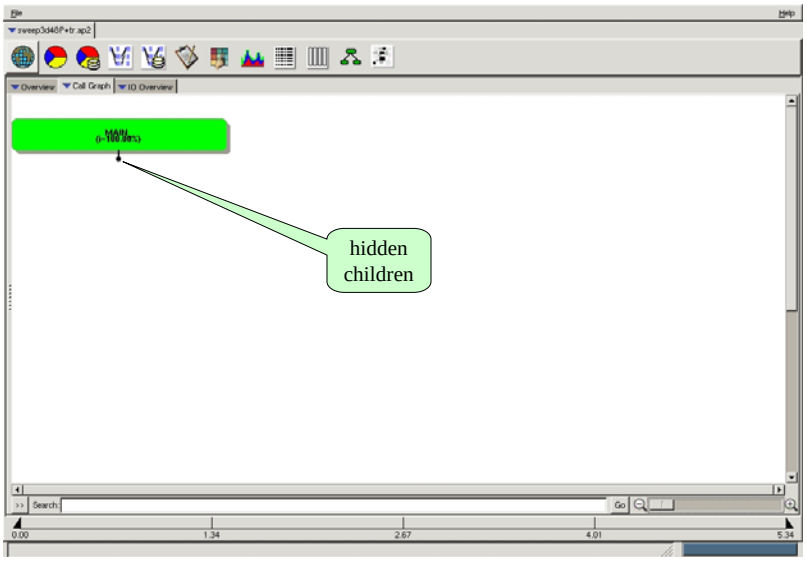
Call Graph View - Zoom



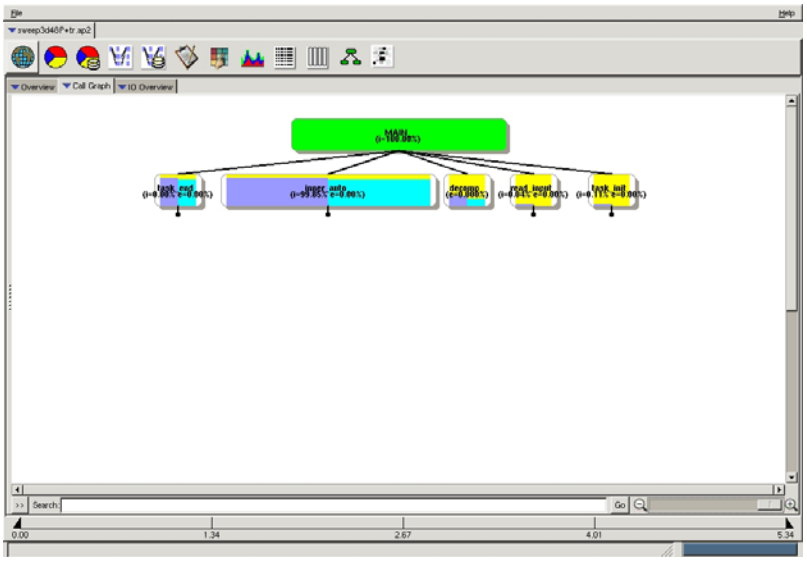
Call Graph View – Function List



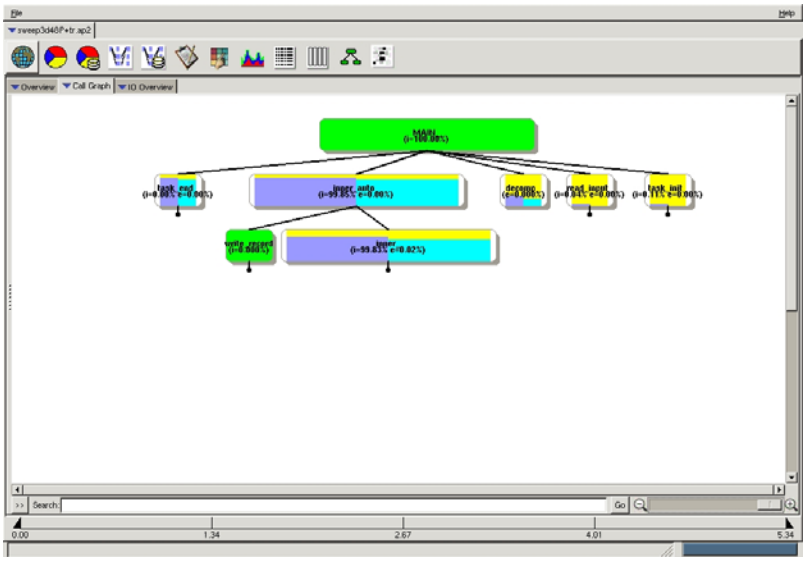
Call Graph Hide Children



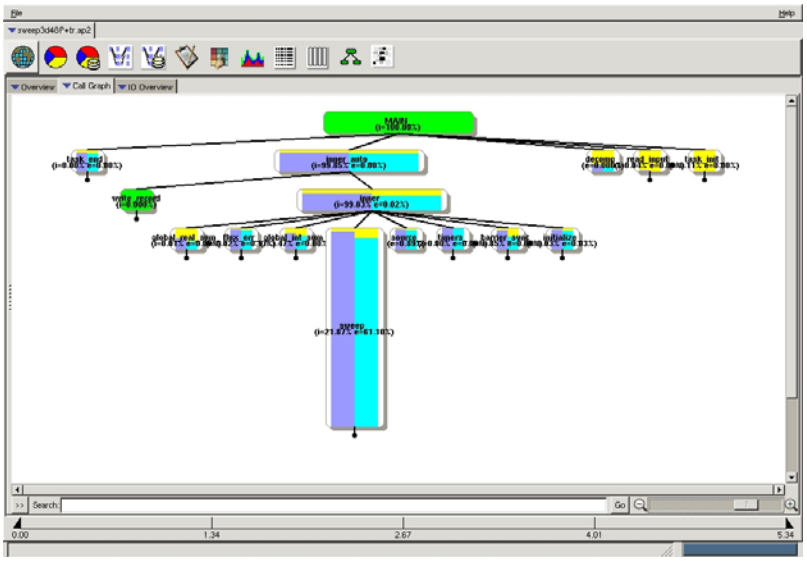
Call Graph Unhide One Level



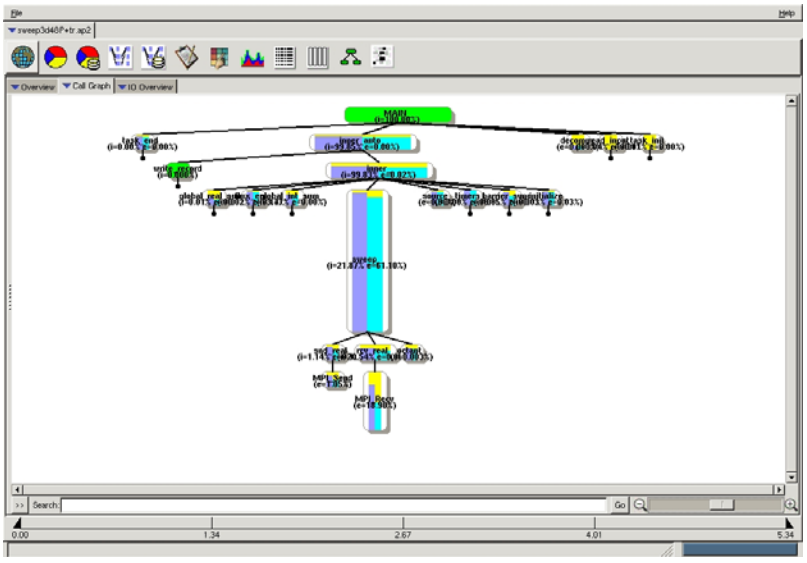
Call Graph Unhide One Level (2)



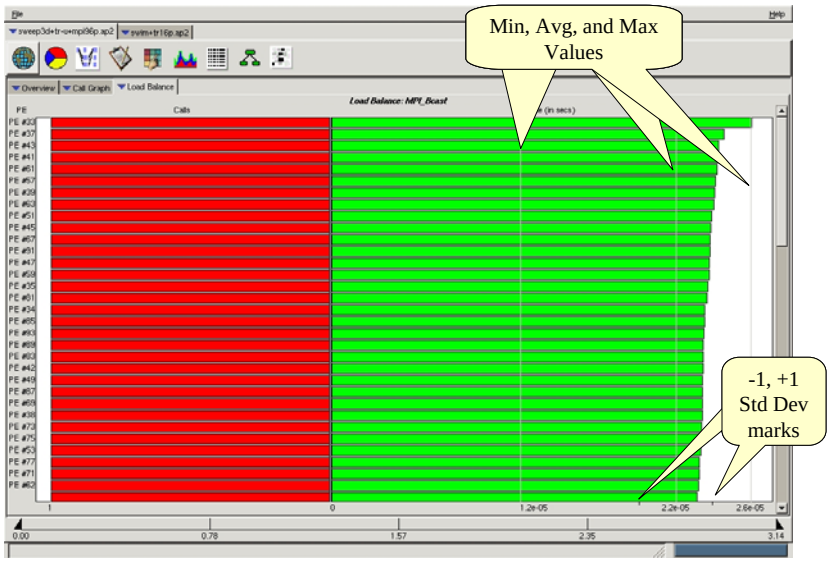
Call Graph Unhide One Level (3)



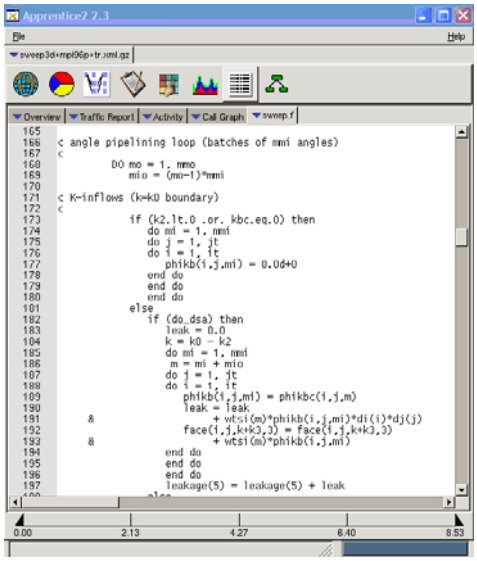
Call Graph Unhide All Children



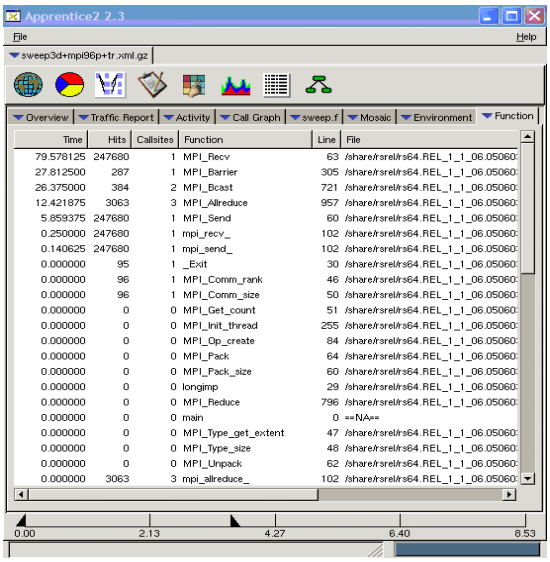
Load Balance View (from Call Graph)



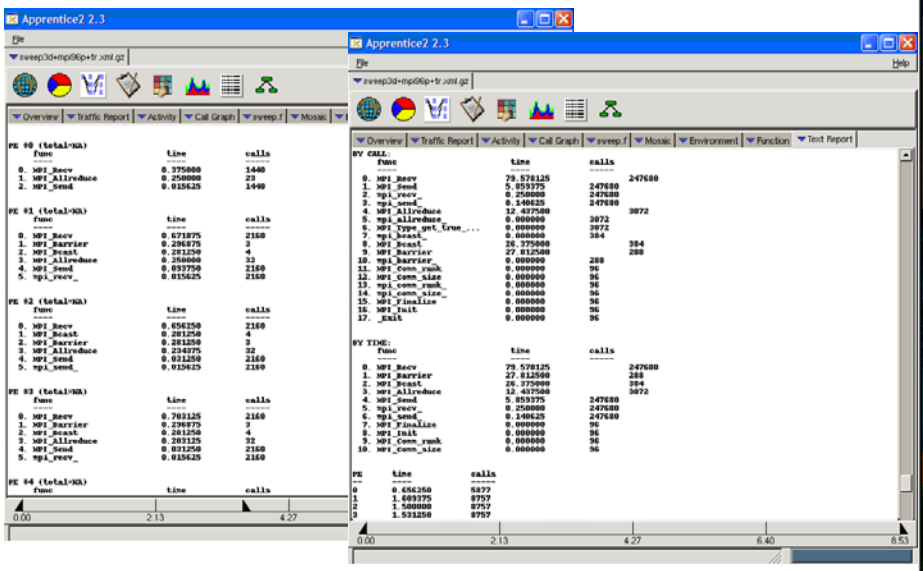
Source Mapping from Call Graph



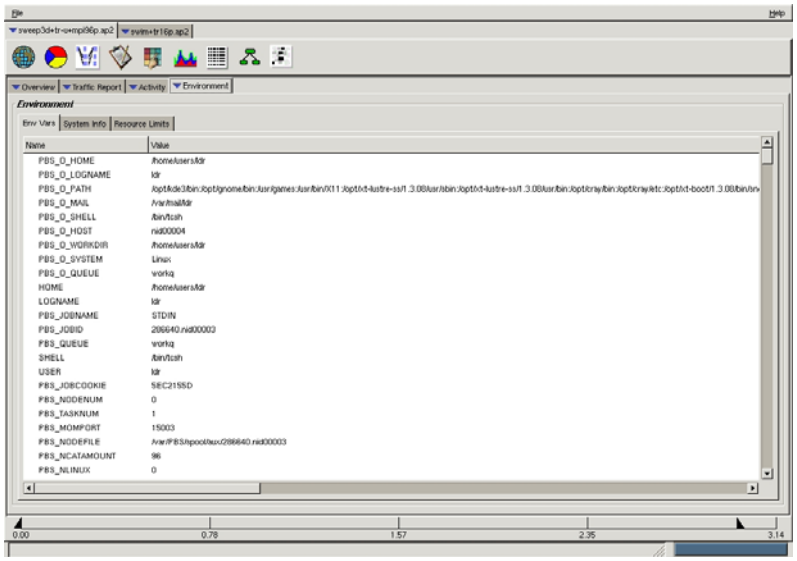
Function Profile and Mapping



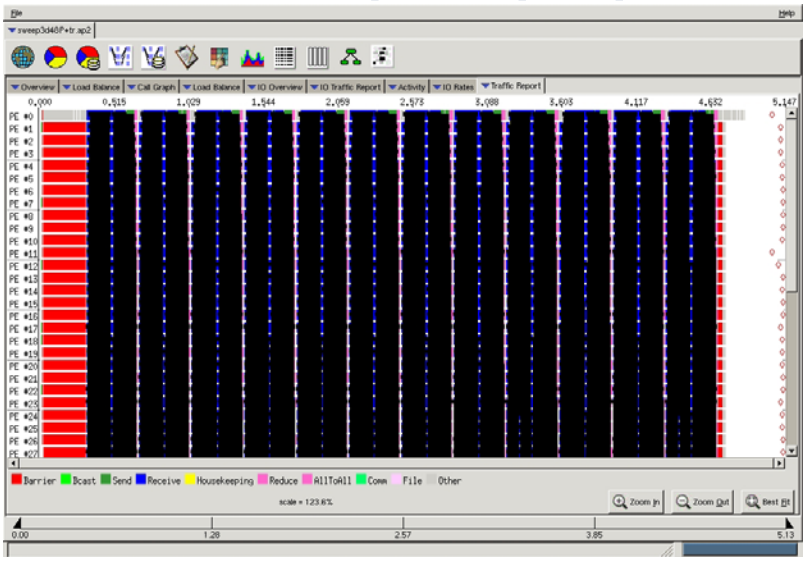
Distribution by PE, by Call, & by Time



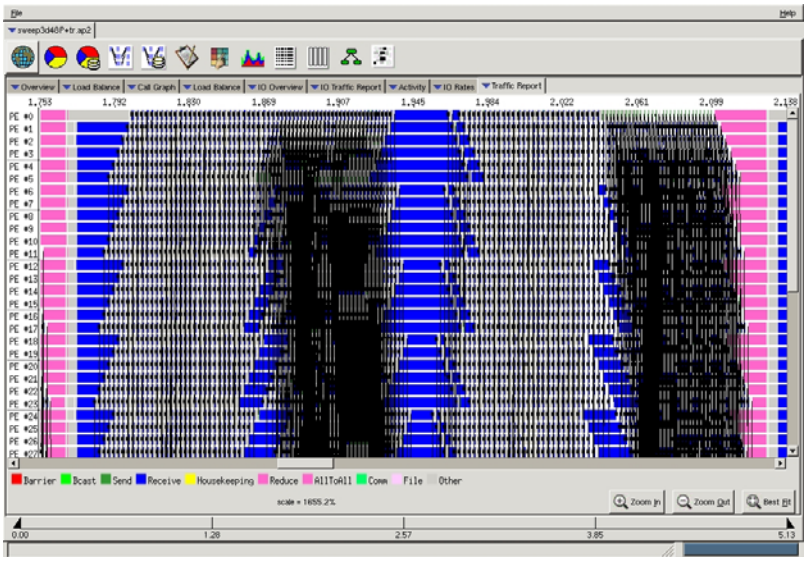
Execution Details



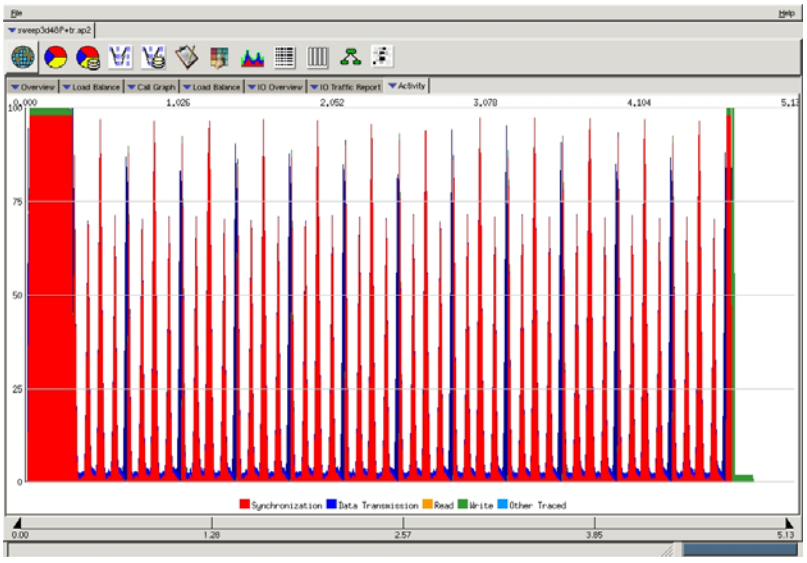
Time Line View (Sweep3D)



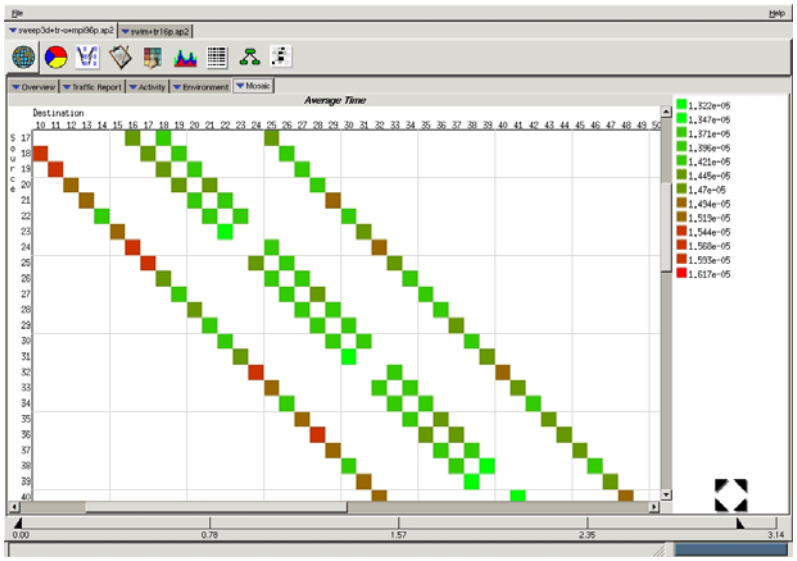
Time Line View (Zoom)



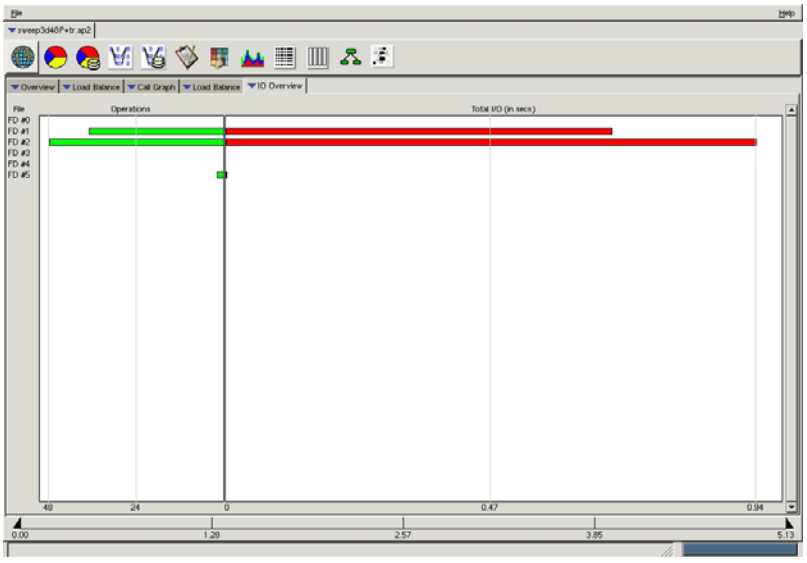
Activity View



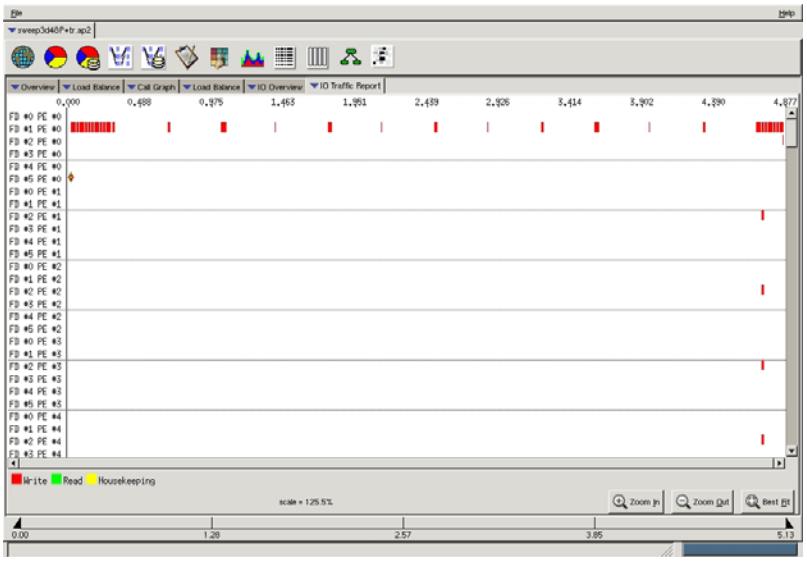
Pair-wise Communication



I/O Overview



I/O Traffic Report



I/O Rates

File Help

sweep3d48P+tr.ap2

Overview | IO Overview | IO Traffic Report | IO Rates

FD	Write Calls	Write Tot (MB)	Write Min (MB/s)	Write Avg (MB/s)	Write Max (MB/s)	Read Calls	Read Tot (MB)	Read Min (MB/s)	Read Avg (MB/s)	Read Max (MB/s)
1	37	0.002	0.001	0.039	0.869	0	0.000	0.000	0.000	0.000
2	48	0.001	0.001	0.003	0.136	0	0.000	0.000	0.000	0.000
5	0	0.000	0.000	0.000	0.000	2	0.000	0.039	0.045	0.051

0.00 1.34 2.67 4.01 5.34

Controlling Trace File Size

- Several environment variables are available to limit trace files to a reasonable size:
  - PAT_RT_CALLSTACK
    - Limit the depth to trace the call stack
  - PAT_RT_HWPC
    - Avoid collecting hardware counters (unset)
  - PAT_RT_RECORD_PE
    - Collect trace for a subset of the PEs
  - PAT_RT_TRACE_FUNCTION_ARGS
    - Limit the number of function arguments to be traced
  - PAT_RT_TRACE_FUNCTION_LIMITS
    - Avoid tracing indicated functions
  - PAT_RT_TRACE_FUNCTION_MAX
    - Limit the maximum number of traces generated for all functions for a single process
- Use the limit built-in command for ksh(1) or csh(1) to control how much disk space the trace file can consume

Performance Measurement, Tuning, and Optimization on the Cray XT3

Questions / Comments
Thank You!

- CUG 2006