Preparing an application for Hybrid Supercomputing using Cray's Tool Suite

John M Levesque
Director – Cray's Supercomputing Center of Excellence

levesque@cray.com

Outline

- Investigate all-MPI application
  - Profile problem of interest
    - perftools-lite, perftools-lite-loops
  - Identification of potential parallel high level loops
    - Reveal
    - Introduction of OpenMP directives
  - Investigation of memory usage – perftools-lite-hbm
  - Moving from OpenMP to OpenMP for accelerators

Using perftools-lite (or -loops or -hbm)

- module load perftools-lite or perftools-lite-loops
  - Module perftools-base should already be loaded
- Build application
- Run application
- Statistics report comes out within standard out
  - Also generates a directory of profile data to be examined with different options

Perftools-lite profile – Run on 8 nodes–8 MPI tasks

```
Notes for table 1:
  This table shows functions that have significant exclusive sample
    hits, averaged across ranks.
  For further explanation, see the "General table notes" below,
    or use:  pat_report -v -O samp_profile ...
Table 1:  Profile by Function (limited entries shown)
  Samp% |      Samp | Imb.  |  Imb.  | Group
        |           | Samp  | Samp%  |  Function=[MAX10]
        |           |       |        |   PE=HIDE
 100.0% | 29,633.0  |   --  |    --  | Total
|-----------------------------------------------------
|  98.5% | 29,183.5 |   --  |    --  | USER
||----------------------------------------------------
||  27.1% |  8,030.6 | 43.4  |  0.6%  | fluxj_
||  26.9% |  7,958.6 | 47.4  |  0.7%  | fluxk_
||  22.8% |  6,745.0 | 79.0  |  1.3%  | fluxi_
||   5.2% |  1,551.4 | 21.6  |  1.6%  | extrapk_
||   5.1% |  1,499.5 | 18.5  |  1.4%  | extrapi_
||   4.6% |  1,376.2 | 24.8  |  2.0%  | update_
||   4.2% |  1,258.9 | 24.1  |  2.1%  | extrapj_
||====================================================
|   1.4% |    407.9 |   --  |    --  | MPI
|====================================================
```

Exclusive time

Sampling is in

100th of a second

Imbalance

Perftools-lite-loops – Run on 8 nodes–8 MPI tasks

Important loops highlighted

```
Table 1:
  Inclusive and Exclusive Time in Loops
  Loop |  Loop Incl |  Loop Hit |  Loop |  Loop |  Loop | Function=/.LOOP[.]
  Incl |       Time |           | Trips | Trips | Trips |  PE=HIDE
 Time% |            |           |   Avg |   Min |   Max |
|-------------------------------------------------------------------------
| 99.4% | 333.895923 |         1 | 500.0 |   500 |   500 | les3d_.LOOP.3.li.216
| 98.7% | 331.721721 |       500 |   2.0 |     2 |     2 | les3d_.LOOP.4.li.272
| 26.5% |  89.032566 |     1,000 |  96.0 |    96 |    96 | fluxk_.LOOP.1.li.28
| 24.6% |  82.681435 |    96,000 |  97.0 |    97 |    97 | fluxk_.LOOP.2.li.29
| 24.2% |  81.356609 |     1,000 |  96.0 |    96 |    96 | fluxj_.LOOP.1.li.28
| 22.5% |  75.770180 |    96,000 |  97.0 |    97 |    97 | fluxj_.LOOP.2.li.29
| 22.5% |  75.458432 |     1,000 |  96.0 |    96 |    96 | fluxi_.LOOP.1.li.21
| 22.5% |  75.453469 |    96,000 |  96.0 |    96 |    96 | fluxi_.LOOP.2.li.22
| 18.9% |  63.574836 | 9,312,000 |  96.0 |    96 |    96 | visck_.LOOP.1.li.344
| 17.1% |  57.529187 | 9,312,000 |  96.0 |    96 |    96 | viscj_.LOOP.1.li.340
| 15.7% |  52.794857 | 9,216,000 |  97.0 |    97 |    97 | visci_.LOOP.1.li.782
|  5.0% |  16.924522 |     1,000 |  99.0 |    99 |    99 | extrapi_.LOOP.1.li.128
|=========================================================================
```

How do I know what the important loops are?

- Pat_report –O calltree < directory produced by perftools-lite-loops run>

- Produces call tree with DO loops included

Perftools-lite loops – Run on 8 nodes–8 MPI tasks

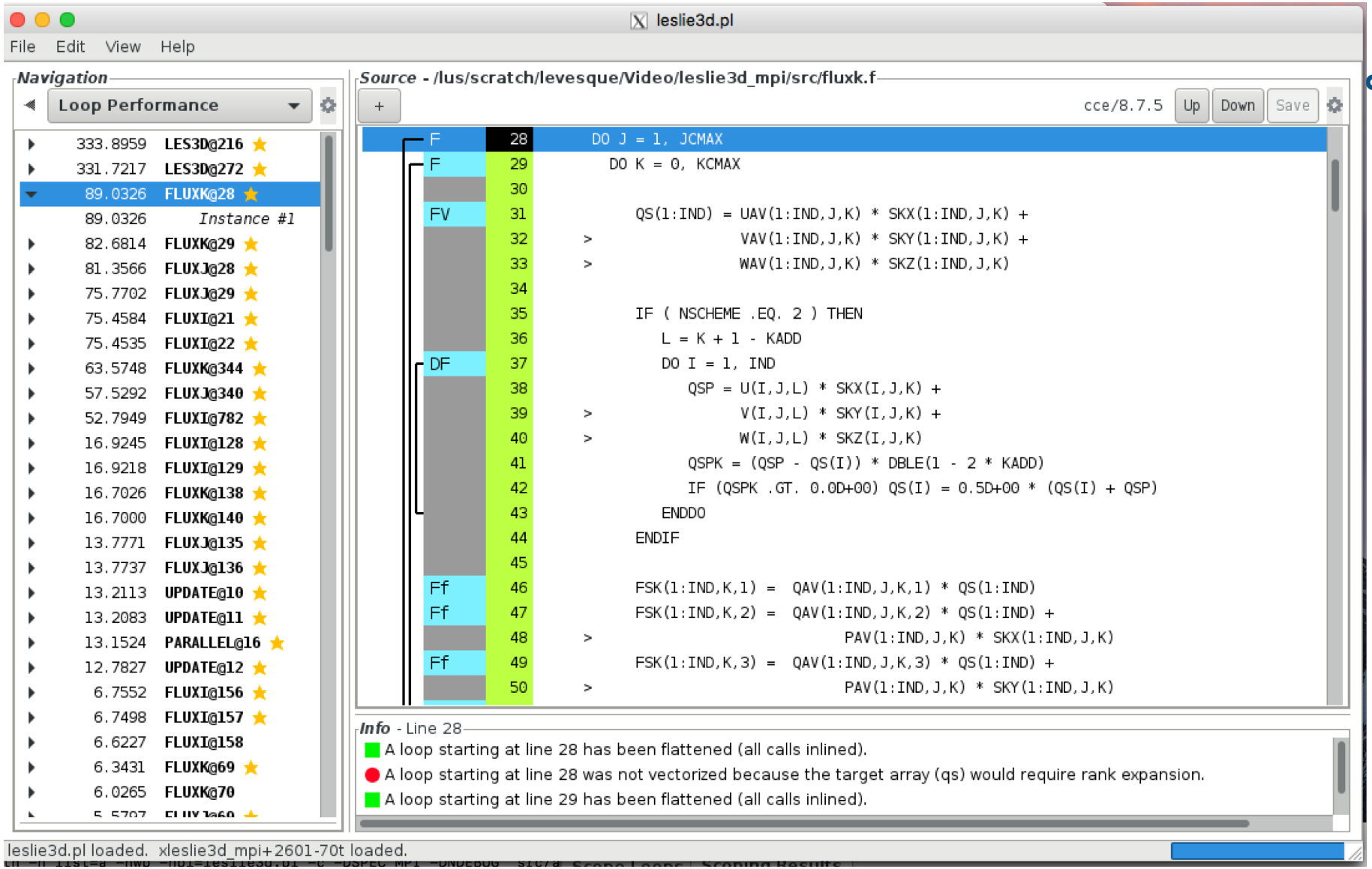
pat_report –O calltree xleslie3d_mpi+2601-70t

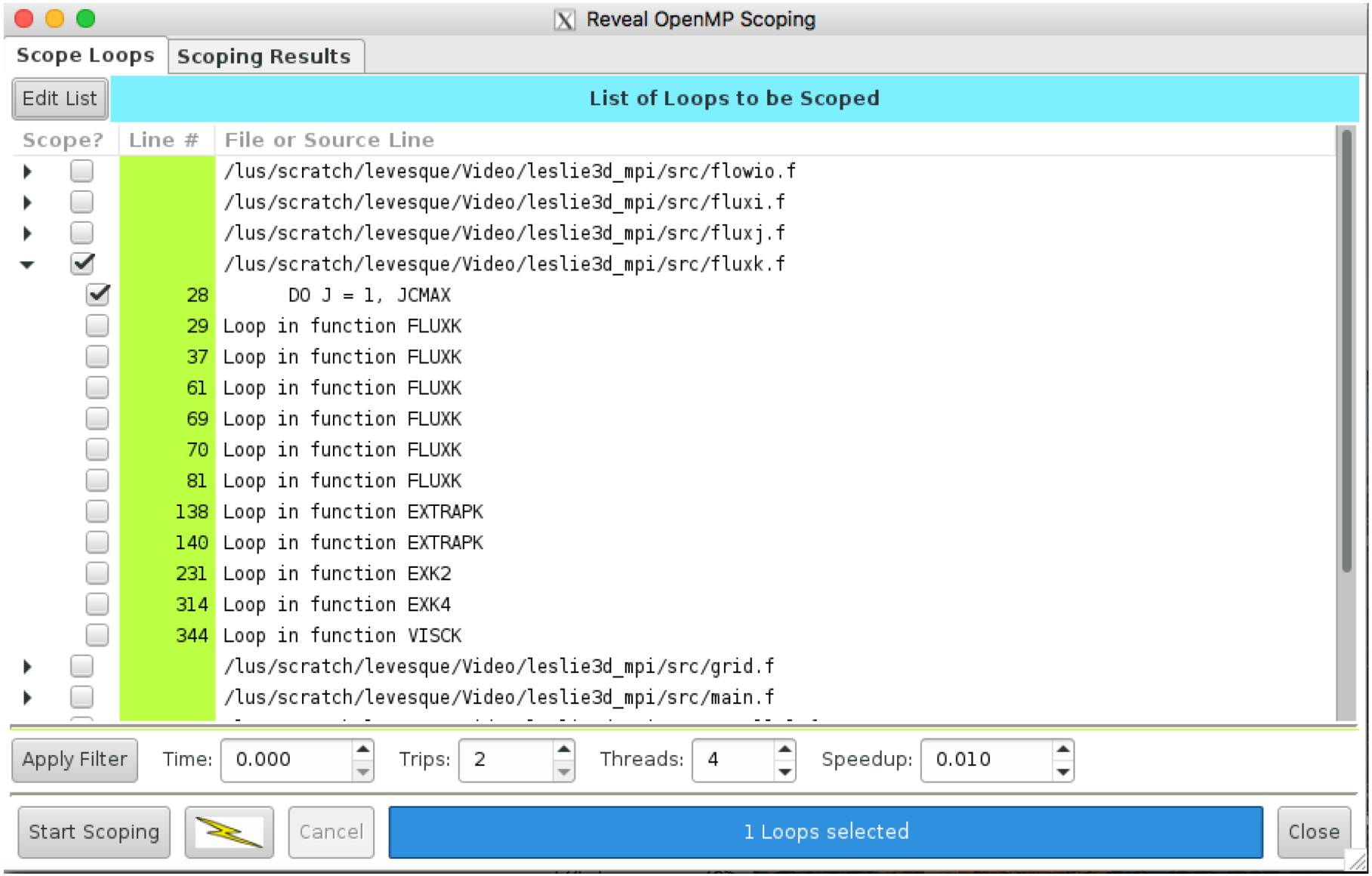
Important loops highlighted

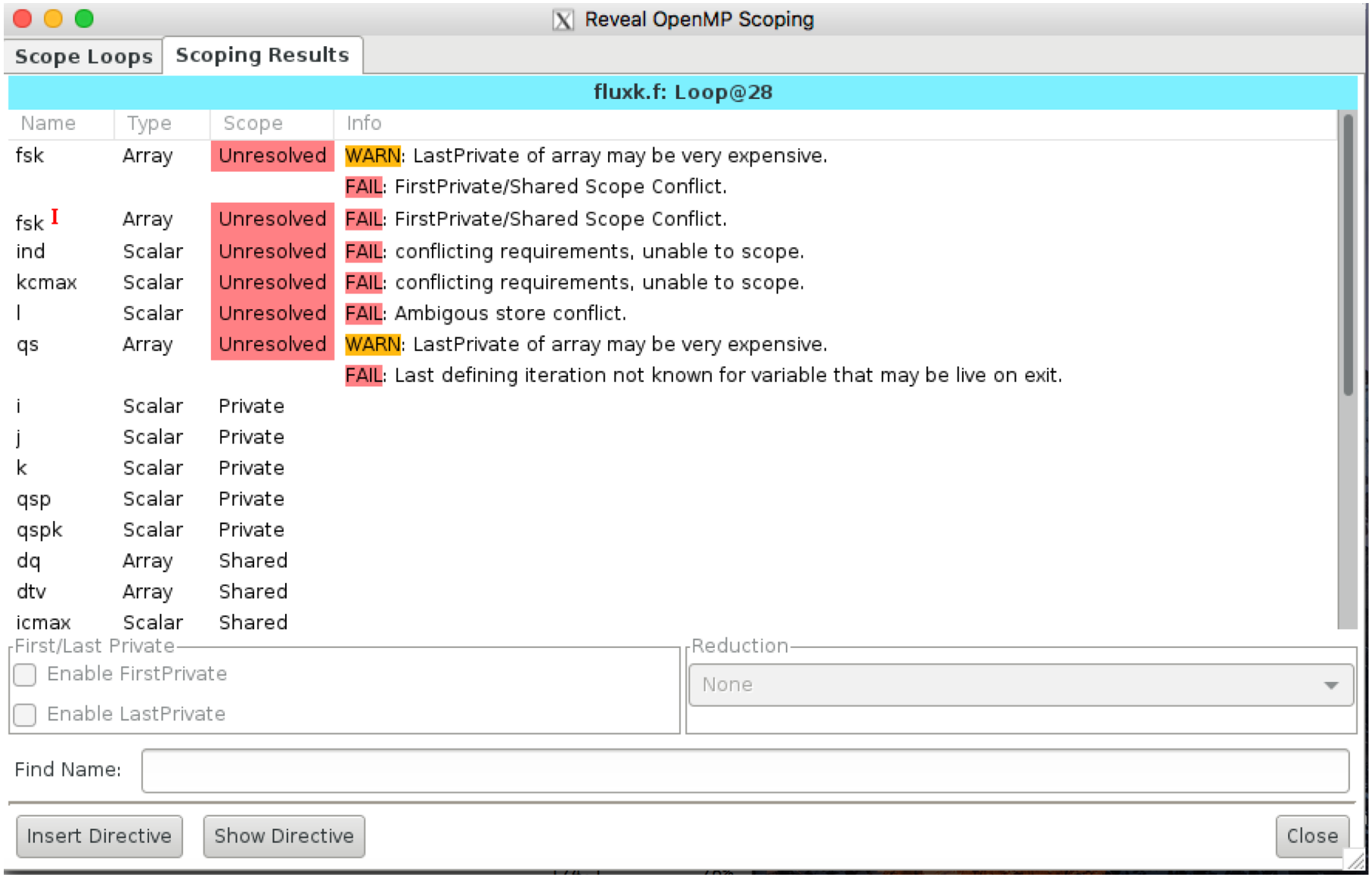
```
Table 1:  Function Calltree View (limited entries shown)
  Time% |       Time |       Calls | Calltree
        |            |             |  PE=HIDE
 100.0% | 331.356157 |          -- | Total
|-----------------------------------------------------------
| 100.0% | 331.356117 |         2.0 | les3d_
|  99.3% | 329.201866 |          -- |  les3d_.LOOP.3.li.216
3  98.7% | 327.029239 |          -- |   les3d_.LOOP.4.li.272
||||-------------------------------------------------------
4|||  31.6% | 104.588754 |     1,000.0 | fluxk_
|||||-----------------------------------------------------
5||||  20.2% |  66.813949 |          -- | fluxk_.LOOP.1.li.28
6||||       |            |              |  fluxk_.LOOP.2.li.29
7||||  20.2% |  66.813949 | 9,312,000.0 |   visck_
5||||   6.4% |  21.068680 |     1,000.0 | fluxk_(exclusive)
5||||   5.0% |  16.706125 |     1,000.0 | extrapk_
|||||=====================================================
4|||  29.2% |  96.783424 |     1,000.0 | fluxi_
|||||-----------------------------------------------------
5||||  16.9% |  55.929620 |          -- | fluxi_.LOOP.1.li.21
6||||       |            |              |  fluxi_.LOOP.2.li.22
7||||  16.9% |  55.929620 | 9,216,000.0 |   visci_
5||||   6.8% |  22.465390 |     1,000.0 | extrapi_
|||||-----------------------------------------------------
6|||||   3.4% |  11.399343 |     1,000.0 | extrapi_(exclusive)
6|||||   3.3% |  11.066047 |          -- | extrapi_.LOOP.1.li.128
7|||||       |            |              |  extrapi_.LOOP.2.li.129
8|||||   3.3% |  11.066047 | 9,801,000.0 |   exi4_
|||||=====================================================
5||||   5.5% |  18.388414 |     1,000.0 | fluxi_(exclusive)
|||||=====================================================
4|||  28.4% |  93.988073 |     1,000.0 | fluxj_
|||||-----------------------------------------------------
5||||  18.3% |  60.687484 |          -- | fluxj_.LOOP.1.li.28
6||||       |            |              |  fluxj_.LOOP.2.li.29
7||||  18.3% |  60.687484 | 9,312,000.0 |   viscj_
5||||   5.9% |  19.520634 |     1,000.0 | fluxj_(exclusive)
5||||   4.2% |  13.779955 |     1,000.0 | extrapj_
|||||=====================================================
4|||   5.3% |  17.668610 |     1,000.0 | parallel_
|||||-----------------------------------------------------
5||||   4.0% |  13.109379 |          -- | parallel_.LOOP.1.li.16
6||||   4.0% |  13.109379 |     5,000.0 |  mpicx_
5||||   1.4% |   4.543585 |     4,000.0 | ghost_
|||||=====================================================
```

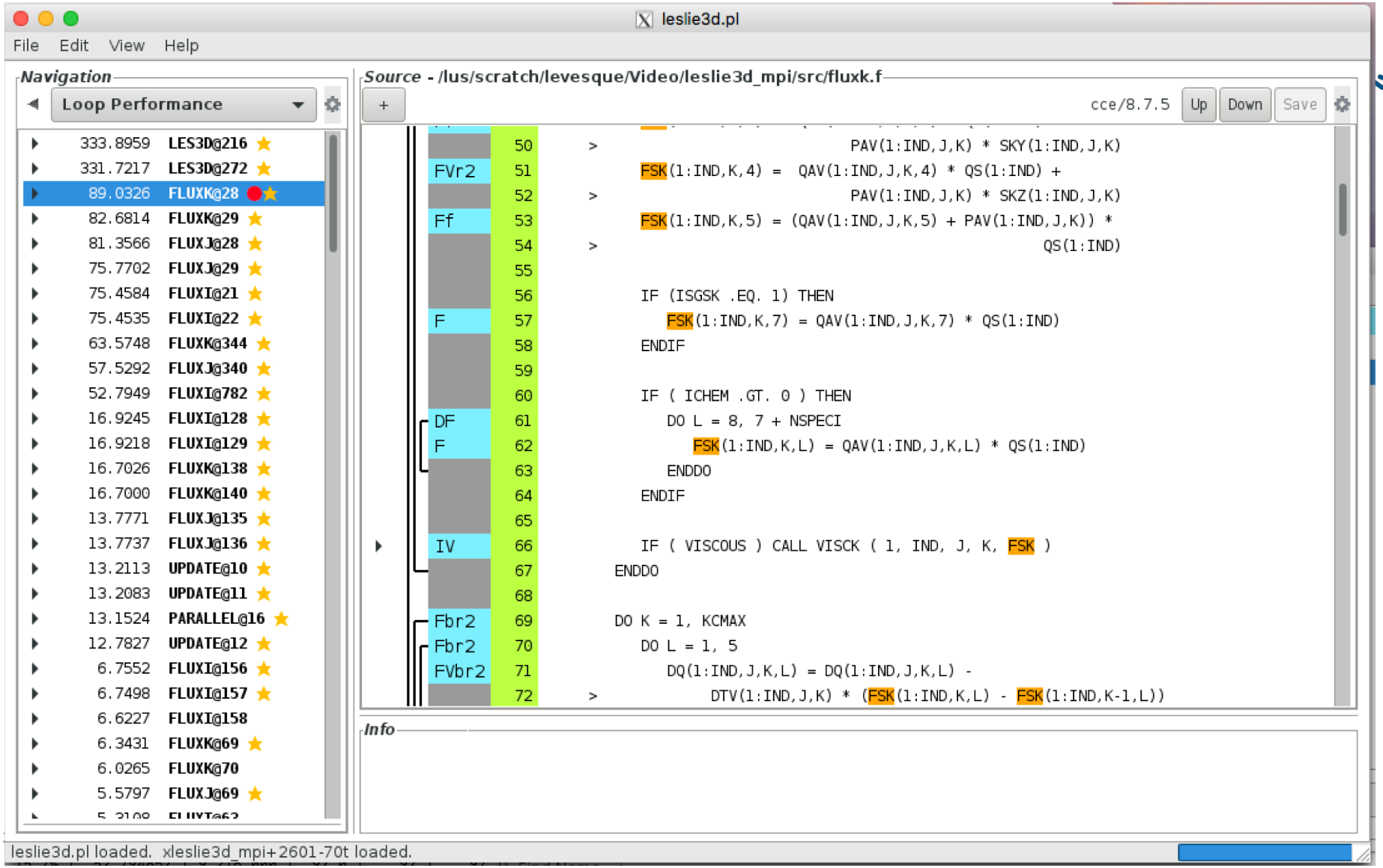
Using Reveal

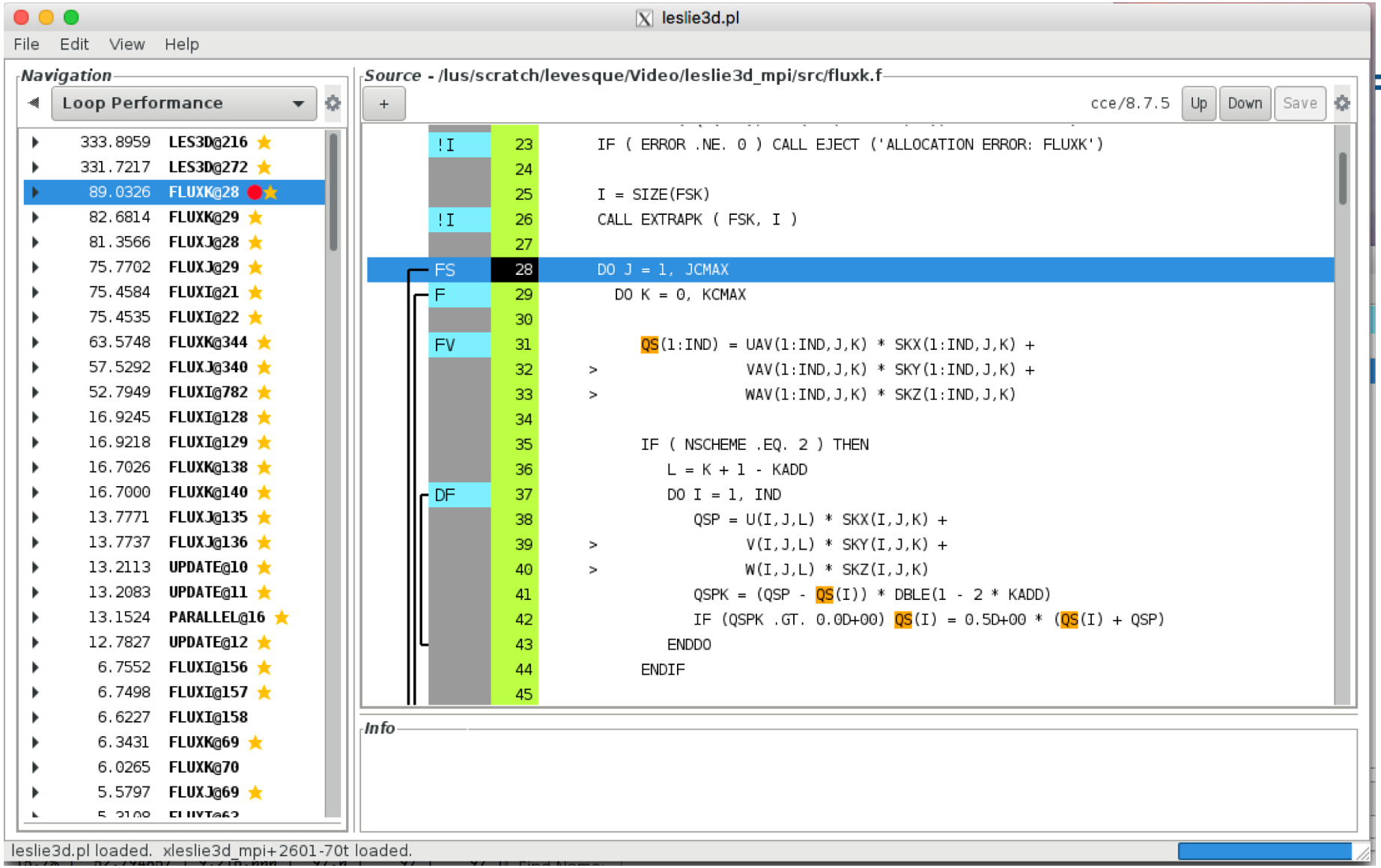
- Need program library and perftools-lite-loops output
  - ftn -hlist=a –hwp –hpl=leslie3d.pl
  - reveal leslie3d.pl < perftools-lite-loops data directory>

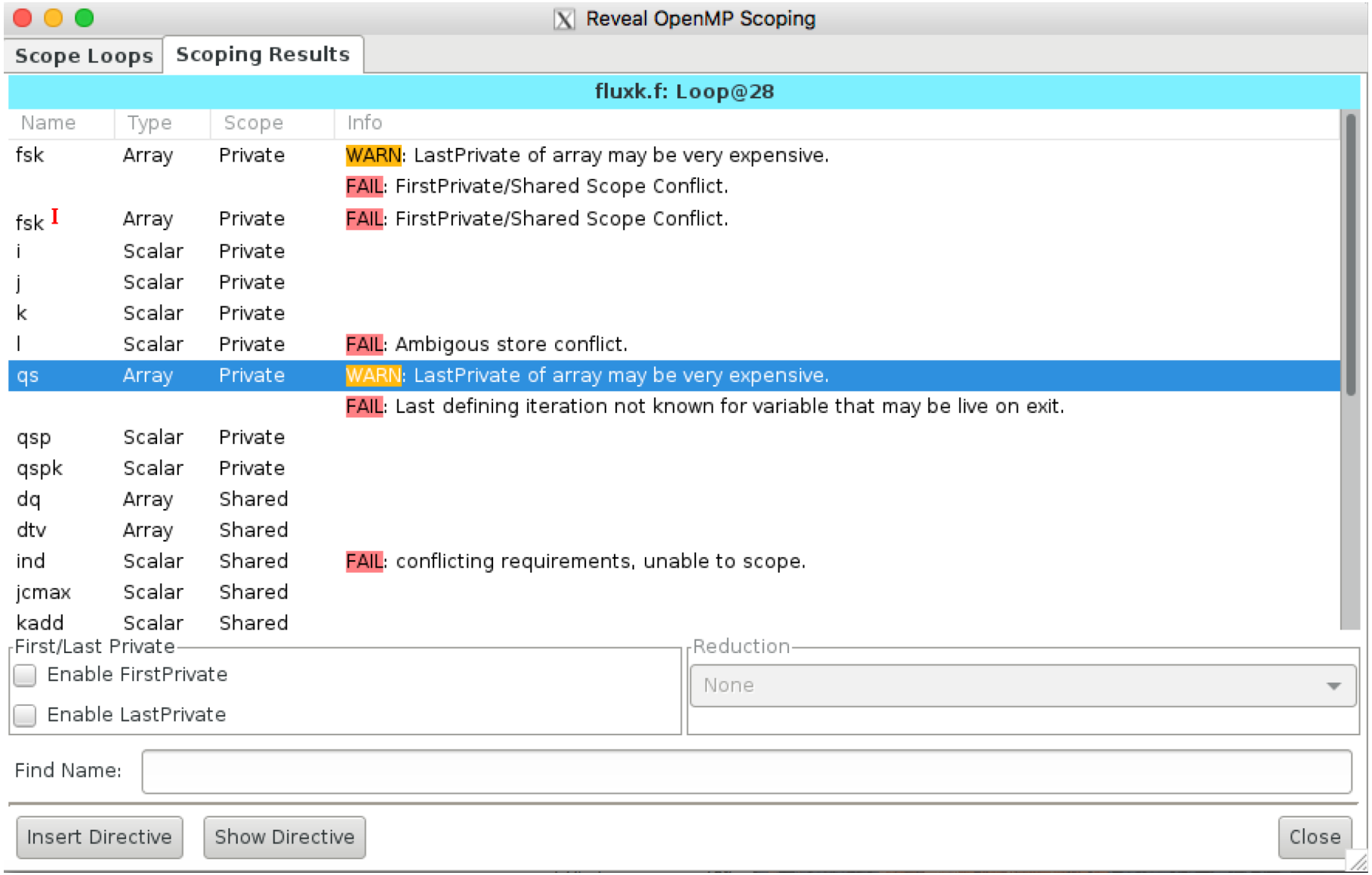

leslie3d.pl
File Edit View Help
Navigation
Loop Performance
333.8959 LES3D@216
331.7217 LES3D@272
89.0326 FLUXK@28
89.0326 Instance #1
82.6814 FLUXK@29
81.3566 FLUXJ@28
75.7702 FLUXJ@29
75.4584 FLUXI@21
75.4535 FLUXI@22
63.5748 FLUXK@344
57.5292 FLUXJ@340
52.7949 FLUXI@782
16.9245 FLUXI@128
16.9218 FLUXI@129
16.7026 FLUXK@138
16.7000 FLUXK@140
13.7771 FLUXJ@135
13.7737 FLUXJ@136
13.2113 UPDATE@10
13.2083 UPDATE@11
13.1524 PARALLEL@16
12.7827 UPDATE@12
6.7552 FLUXI@156
6.7498 FLUXI@157
6.6227 FLUXI@158
6.3431 FLUXK@69
6.0265 FLUXK@70
Source - /lus/scratch/levesque/Video/leslie3d_mpi/src/fluxk.f
cce/8.7.5
Up
Down
Save
F 28 DO J = 1, JCMAX
F 29 DO K = 0, KMAX
30
FV 31 QS(1:IND) = UAV(1:IND,J,K) * SKX(1:IND,J,K) +
32 > VAV(1:IND,J,K) * SKY(1:IND,J,K) +
33 > WAV(1:IND,J,K) * SKZ(1:IND,J,K)
34
35 IF (NSCHEME .EQ. 2) THEN
36 L = K + 1 - KADD
DF 37 DO I = 1, IND
38 QSP = U(I,J,L) * SKX(I,J,K) +
39 > V(I,J,L) * SKY(I,J,K) +
40 > W(I,J,L) * SKZ(I,J,K)
41 QSPK = (QSP - QS(I)) * DBLE(1 - 2 * KADD)
42 IF (QSPK .GT. 0.0D+00) QS(I) = 0.5D+00 * (QS(I) + QSP)
43 ENDDO
44 ENDIF
45
Ff 46 FSK(1:IND,K,1) = QAV(1:IND,J,K,1) * QS(1:IND)
Ff 47 FSK(1:IND,K,2) = QAV(1:IND,J,K,2) * QS(1:IND) +
48 > PAV(1:IND,J,K) * SKX(1:IND,J,K)
Ff 49 FSK(1:IND,K,3) = QAV(1:IND,J,K,3) * QS(1:IND) +
50 > PAV(1:IND,J,K) * SKY(1:IND,J,K)
Info - Line 28
A loop starting at line 28 has been flattened (all calls inlined).
A loop starting at line 28 was not vectorized because the target array (qs) would require rank expansion.
A loop starting at line 29 has been flattened (all calls inlined).
leslie3d.pl loaded. xleslie3d_mpi+2601-70t loaded.

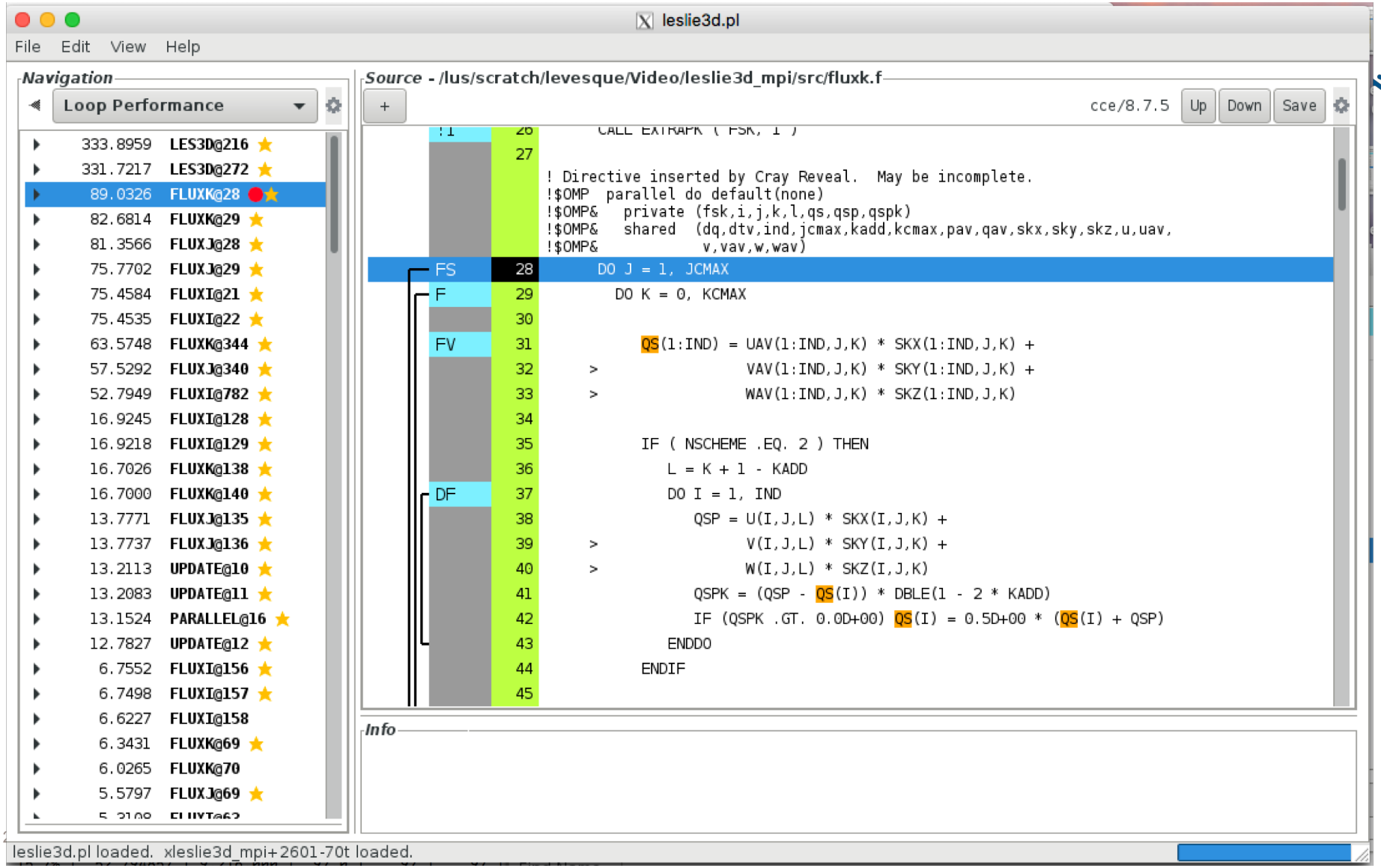

Reveal OpenMP Scoping
Scope Loops
Scoping Results
Edit List
List of Loops to be Scoped
Scope?
Line #
File or Source Line
/lus/scratch/levesque/Video/leslie3d_mpi/src/flowio.f
/lus/scratch/levesque/Video/leslie3d_mpi/src/fluxi.f
/lus/scratch/levesque/Video/leslie3d_mpi/src/fluxj.f
/lus/scratch/levesque/Video/leslie3d_mpi/src/fluxk.f
28 DO J = 1, JCMAX
29 Loop in function FLUXK
37 Loop in function FLUXK
61 Loop in function FLUXK
69 Loop in function FLUXK
70 Loop in function FLUXK
81 Loop in function FLUXK
138 Loop in function EXTRAPK
140 Loop in function EXTRAPK
231 Loop in function EXK2
314 Loop in function EXK4
344 Loop in function VISCK
/lus/scratch/levesque/Video/leslie3d_mpi/src/grid.f
/lus/scratch/levesque/Video/leslie3d_mpi/src/main.f
Apply Filter
Time: 0.000
Trips: 2
Threads: 4
Speedup: 0.010
Start Scoping
Cancel
1 Loops selected
Close


Reveal OpenMP Scoping
Scope Loops
Scoping Results
fluxk.f: Loop@28
Name
Type
Scope
Info
fsk
Array
Unresolved
WARN: LastPrivate of array may be very expensive.
FAIL: FirstPrivate/Shared Scope Conflict.
fsk
Array
Unresolved
FAIL: FirstPrivate/Shared Scope Conflict.
ind
Scalar
Unresolved
FAIL: conflicting requirements, unable to scope.
kcmax
Scalar
Unresolved
FAIL: conflicting requirements, unable to scope.
l
Scalar
Unresolved
FAIL: Ambigous store conflict.
qs
Array
Unresolved
WARN: LastPrivate of array may be very expensive.
FAIL: Last defining iteration not known for variable that may be live on exit.
i
Scalar
Private
j
Scalar
Private
k
Scalar
Private
qsp
Scalar
Private
qspk
Scalar
Private
dq
Array
Shared
dtv
Array
Shared
icmax
Scalar
Shared
First/Last Private
Enable FirstPrivate
Enable LastPrivate
Reduction
None
Find Name:
Insert Directive
Show Directive
Close


leslie3d.pl
File Edit View Help
Navigation
Loop Performance
333.8959 LES3D@216
331.7217 LES3D@272
89.0326 FLUXK@28
82.6814 FLUXK@29
81.3566 FLUXJ@28
75.7702 FLUXJ@29
75.4584 FLUXI@21
75.4535 FLUXI@22
63.5748 FLUXK@344
57.5292 FLUXJ@340
52.7949 FLUXI@782
16.9245 FLUXI@128
16.9218 FLUXI@129
16.7026 FLUXK@138
16.7000 FLUXK@140
13.7771 FLUXJ@135
13.7737 FLUXJ@136
13.2113 UPDATE@10
13.2083 UPDATE@11
13.1524 PARALLEL@16
12.7827 UPDATE@12
6.7552 FLUXI@156
6.7498 FLUXI@157
6.6227 FLUXI@158
6.3431 FLUXK@69
6.0265 FLUXK@70
5.5797 FLUXJ@69
Source - /lus/scratch/levesque/Video/leslie3d_mpi/src/fluxk.f
+
cce/8.7.5
Up
Down
Save
50 > PAV(1:IND,J,K) * SKY(1:IND,J,K)
FVr2 51 FSK(1:IND,K,4) = QAV(1:IND,J,K,4) * QS(1:IND) +
52 > PAV(1:IND,J,K) * SKZ(1:IND,J,K)
Ff 53 FSK(1:IND,K,5) = (QAV(1:IND,J,K,5) + PAV(1:IND,J,K)) *
54 > QS(1:IND)
55
56 IF (ISGSK .EQ. 1) THEN
F 57 FSK(1:IND,K,7) = QAV(1:IND,J,K,7) * QS(1:IND)
58 ENDIF
59
60 IF (ICHEM .GT. 0) THEN
DF 61 DO L = 8, 7 + NSPECI
F 62 FSK(1:IND,K,L) = QAV(1:IND,J,K,L) * QS(1:IND)
63 ENDDO
64 ENDIF
65
IV 66 IF (VISCOUS) CALL VISCK (1, IND, J, K, FSK)
67 ENDDO
68
Fbr2 69 DO K = 1, KCMAX
Fbr2 70 DO L = 1, 5
FVbr2 71 DQ(1:IND,J,K,L) = DQ(1:IND,J,K,L) -
72 > DTV(1:IND,J,K) * (FSK(1:IND,K,L) - FSK(1:IND,K-1,L))
Info
leslie3d.pl loaded. xleslie3d_mpi+2601-70t loaded.


leslie3d.pl
File Edit View Help
Navigation
Loop Performance
333.8959 LES3D@216
331.7217 LES3D@272
89.0326 FLUXK@28
82.6814 FLUXK@29
81.3566 FLUXJ@28
75.7702 FLUXJ@29
75.4584 FLUXI@21
75.4535 FLUXI@22
63.5748 FLUXK@344
57.5292 FLUXJ@340
52.7949 FLUXI@782
16.9245 FLUXI@128
16.9218 FLUXI@129
16.7026 FLUXK@138
16.7000 FLUXK@140
13.7771 FLUXJ@135
13.7737 FLUXJ@136
13.2113 UPDATE@10
13.2083 UPDATE@11
13.1524 PARALLEL@16
12.7827 UPDATE@12
6.7552 FLUXI@156
6.7498 FLUXI@157
6.6227 FLUXI@158
6.3431 FLUXK@69
6.0265 FLUXK@70
5.5797 FLUXJ@69
Source - /lus/scratch/levesque/Video/leslie3d_mpi/src/fluxk.f
cce/8.7.5
Up
Down
Save
!I 23 IF (ERROR .NE. 0) CALL EJECT ('ALLOCATION ERROR: FLUXK')
24
25 I = SIZE(FSK)
!I 26 CALL EXTRAPK (FSK, I)
27
FS 28 DO J = 1, JCMAX
F 29 DO K = 0, KCMAX
30
FV 31 QS(1:IND) = UAV(1:IND,J,K) * SKX(1:IND,J,K) +
32 > VAV(1:IND,J,K) * SKY(1:IND,J,K) +
33 > WAV(1:IND,J,K) * SKZ(1:IND,J,K)
34
35 IF (NSCHEME .EQ. 2) THEN
36 L = K + 1 - KADD
DF 37 DO I = 1, IND
38 QSP = U(I,J,L) * SKX(I,J,K) +
39 > V(I,J,L) * SKY(I,J,K) +
40 > W(I,J,L) * SKZ(I,J,K)
41 QSPK = (QSP - QS(I)) * DBLE(1 - 2 * KADD)
42 IF (QSPK .GT. 0.0D+00) QS(I) = 0.5D+00 * (QS(I) + QSP)
43 ENDDO
44 ENDIF
45
Info
leslie3d.pl loaded. xleslie3d_mpi+2601-70t loaded.


Reveal OpenMP Scoping
Scope Loops
Scoping Results
fluxk.f: Loop@28
Name Type Scope Info
fsk Array Private WARN: LastPrivate of array may be very expensive.
FAIL: FirstPrivate/Shared Scope Conflict.
fsk Array Private FAIL: FirstPrivate/Shared Scope Conflict.
i Scalar Private
j Scalar Private
k Scalar Private
l Scalar Private FAIL: Ambigous store conflict.
qs Array Private WARN: LastPrivate of array may be very expensive.
FAIL: Last defining iteration not known for variable that may be live on exit.
qsp Scalar Private
qspk Scalar Private
dq Array Shared
dtv Array Shared
ind Scalar Shared FAIL: conflicting requirements, unable to scope.
jcmax Scalar Shared
kadd Scalar Shared
First/Last Private
Enable FirstPrivate
Enable LastPrivate
Reduction
None
Find Name:
Insert Directive
Show Directive
Close

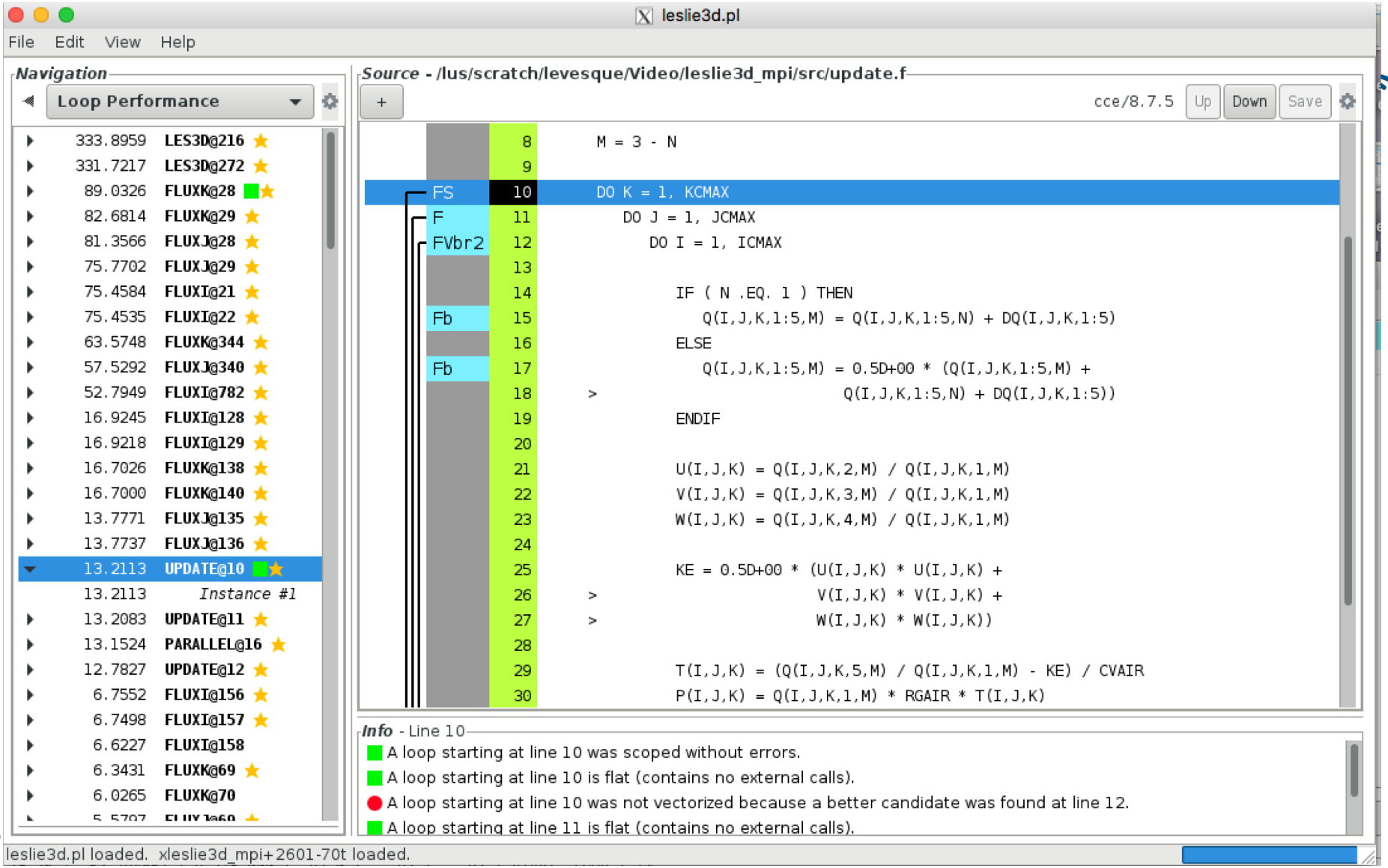

leslie3d.pl
File Edit View Help
Navigation
Loop Performance
333.8959 LES3D@216
331.7217 LES3D@272
89.0326 FLUXK@28
82.6814 FLUXK@29
81.3566 FLUXJ@28
75.7702 FLUXJ@29
75.4584 FLUXI@21
75.4535 FLUXI@22
63.5748 FLUXK@344
57.5292 FLUXJ@340
52.7949 FLUXI@782
16.9245 FLUXI@128
16.9218 FLUXI@129
16.7026 FLUXK@138
16.7000 FLUXK@140
13.7771 FLUXJ@135
13.7737 FLUXJ@136
13.2113 UPDATE@10
13.2083 UPDATE@11
13.1524 PARALLEL@16
12.7827 UPDATE@12
6.7552 FLUXI@156
6.7498 FLUXI@157
6.6227 FLUXI@158
6.3431 FLUXK@69
6.0265 FLUXK@70
5.5797 FLUXJ@69
Source - /lus/scratch/levesque/Video/leslie3d_mpi/src/fluxk.f
cce/8.7.5
Up
Down
Save
27
! Directive inserted by Cray Reveal. May be incomplete.
!$OMP parallel do default(none)
!$OMP& private (fsk,i,j,k,l,qs,qsp,qspk)
!$OMP& shared (dq,dtv,ind,jcmax,kadd,kcmax,pav,qav,skx,sky,skz,u,uav,
!$OMP& v,vav,w,wav)
FS 28 DO J = 1, JCMAX
F 29 DO K = 0, KCMAX
30
FV 31 QS(1:IND) = UAV(1:IND,J,K) * SKX(1:IND,J,K) +
32 > VAV(1:IND,J,K) * SKY(1:IND,J,K) +
33 > WAV(1:IND,J,K) * SKZ(1:IND,J,K)
34
35 IF (NSCHEME .EQ. 2) THEN
36 L = K + 1 - KADD
DF 37 DO I = 1, IND
38 QSP = U(I,J,L) * SKX(I,J,K) +
39 > V(I,J,L) * SKY(I,J,K) +
40 > W(I,J,L) * SKZ(I,J,K)
41 QSPK = (QSP - QS(I)) * DBLE(1 - 2 * KADD)
42 IF (QSPK .GT. 0.0D+00) QS(I) = 0.5D+00 * (QS(I) + QSP)
43 ENDDO
44 ENDIF
45
Info
leslie3d.pl loaded. xleslie3d_mpi+2601-70t loaded.

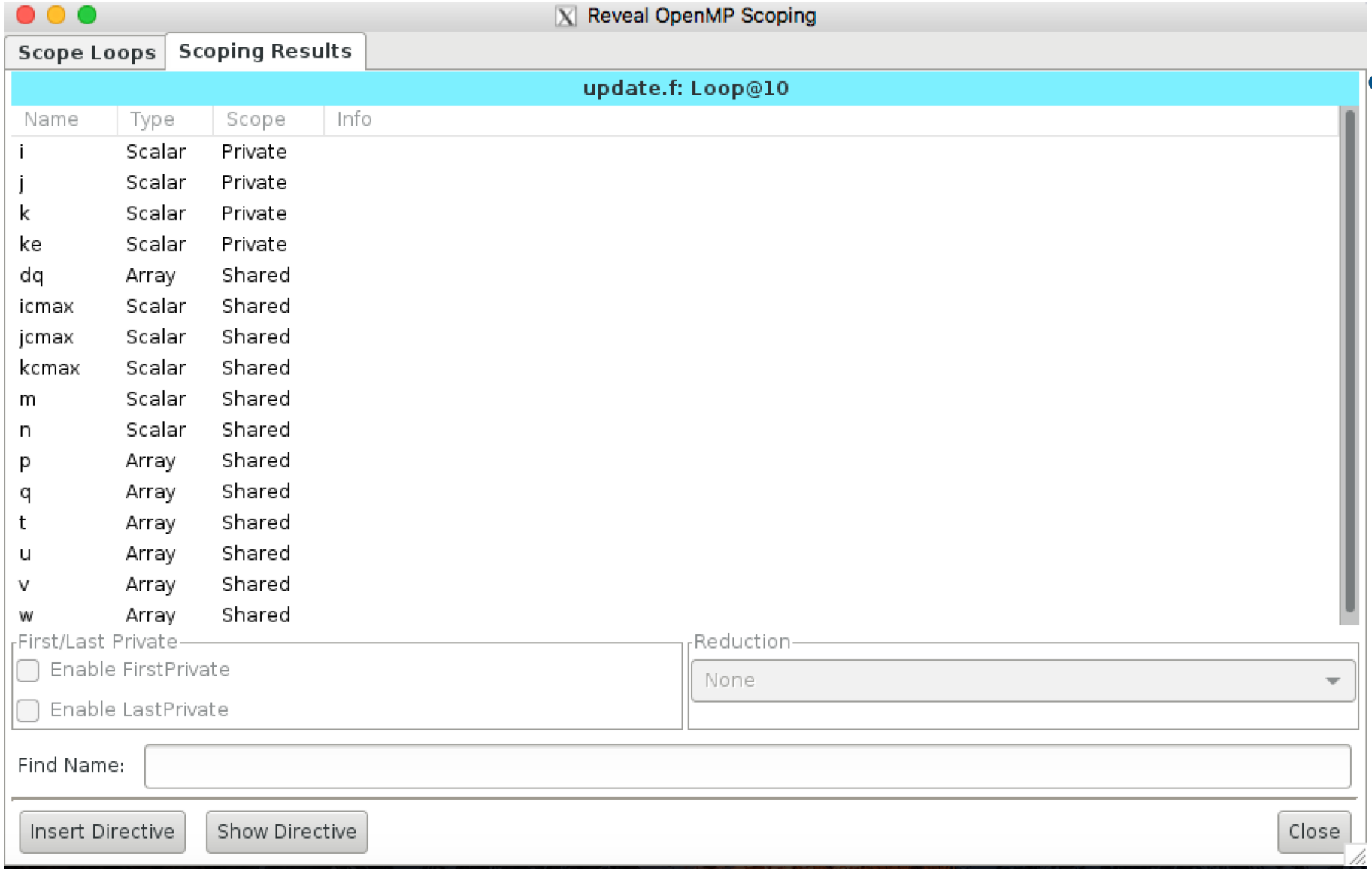
Reveal helped analyze this loop

```
28.                        ! Directive inserted by Cray Reveal.  May be incomplete.
29.    M-----------------< !$OMP  parallel do default(none)
30.    M                   !$OMP&   private (fsk,i,j,k,l,qs,qsp,qspk)
31.    M                   !$OMP&   shared  (dq,dtv,ind,jcmax,kadd,kcmax,pav,qav,skx,sky,skz,u,uav,
32.    M                   !$OMP&            v,vav,w,wav)
33.  + M mF--------------<       DO J = 1, JCMAX
34.  + M mF F------------<         DO K = 0, KCMAX
35.    M mF F V----------<          DO I = 1, IND
36.    M mF F V                      QS(I) = UAV(I,J,K) * SKX(I,J,K) +
37.    M mF F V                >                VAV(I,J,K) * SKY(I,J,K) +
38.    M mF F V                >                WAV(I,J,K) * SKZ(I,J,K)
39.    M mF F V
40.    M mF F V                      IF ( NSCHEME .EQ. 2 ) THEN
41.    M mF F V                         L = K + 1 - KADD
42.    M mF F V                         QSP = U(I,J,L) * SKX(I,J,K) +
43.    M mF F V                >              V(I,J,L) * SKY(I,J,K) +
44.    M mF F V                >              W(I,J,L) * SKZ(I,J,K)
45.    M mF F V                         QSPK = (QSP - QS(I)) * DBLE(1 - 2 * KADD)
46.    M mF F V                         IF (QSPK .GT. 0.0D+00) QS(I) = 0.5D+00 * (QS(I) + QSP)
47.    M mF F V                      ENDIF
48.    M mF F V
49.    M mF F V                      FSK(I,K,1) =  QAV(I,J,K,1) * QS(I)
50.    M mF F V                      FSK(I,K,2) =  QAV(I,J,K,2) * QS(I) +
51.    M mF F V                >                            PAV(I,J,K) * SKX(I,J,K)
52.    M mF F V                      FSK(I,K,3) =  QAV(I,J,K,3) * QS(I) +
53.    M mF F V                >                            PAV(I,J,K) * SKY(I,J,K)
54.    M mF F V                      FSK(I,K,4) =  QAV(I,J,K,4) * QS(I) +
55.    M mF F V                >                            PAV(I,J,K) * SKZ(I,J,K)
56.    M mF F V                      FSK(I,K,5) = (QAV(I,J,K,5) + PAV(I,J,K)) *
57.    M mF F V                >                                                 QS(I)
```

Reveal helped analyze this loop

```
58.     M mF F V
59.     M mF F V                        IF (ISGSK .EQ. 1) THEN
60.     M mF F V                           FSK(I,K,7) = QAV(I,J,K,7) * QS(I)
61.     M mF F V                        ENDIF
62.     M mF F V
63.     M mF F V                        IF ( ICHEM .GT. 0 ) THEN
64.     M mF F V D--------<                DO L = 8, 7 + NSPECI
65.     M mF F V D                            FSK(I,K,L) = QAV(I,J,K,L) * QS(I)
66.     M mF F V D-------->                ENDDO
67.     M mF F V                        ENDIF
68.     M mF F V---------->          ENDDO
69.     M mF F
70.   + M mF F V I-------<>             IF ( VISCOUS ) CALL VISCK ( 1, IND, J, K, FSK )
71.     M mF F------------>          ENDDO
```


leslie3d.pl
File Edit View Help
Navigation
Loop Performance
333.8959 LES3D@216
331.7217 LES3D@272
89.0326 FLUXK@28
82.6814 FLUXK@29
81.3566 FLUXJ@28
75.7702 FLUXJ@29
75.4584 FLUXI@21
75.4535 FLUXI@22
63.5748 FLUXK@344
57.5292 FLUXJ@340
52.7949 FLUXI@782
16.9245 FLUXI@128
16.9218 FLUXI@129
16.7026 FLUXK@138
16.7000 FLUXK@140
13.7771 FLUXJ@135
13.7737 FLUXJ@136
13.2113 UPDATE@10
13.2113 Instance #1
13.2083 UPDATE@11
13.1524 PARALLEL@16
12.7827 UPDATE@12
6.7552 FLUXI@156
6.7498 FLUXI@157
6.6227 FLUXI@158
6.3431 FLUXK@69
6.0265 FLUXK@70
Source - /lus/scratch/levesque/Video/leslie3d_mpi/src/update.f
+
cce/8.7.5
Up
Down
Save
8 M = 3 - N
9
FS 10 DO K = 1, KCMAX
F 11 DO J = 1, JCMAX
FVbr2 12 DO I = 1, ICMAX
13
14 IF (N .EQ. 1) THEN
Fb 15 Q(I,J,K,1:5,M) = Q(I,J,K,1:5,N) + DQ(I,J,K,1:5)
16 ELSE
Fb 17 Q(I,J,K,1:5,M) = 0.5D+00 * (Q(I,J,K,1:5,M) +
18 > Q(I,J,K,1:5,N) + DQ(I,J,K,1:5))
19 ENDIF
20
21 U(I,J,K) = Q(I,J,K,2,M) / Q(I,J,K,1,M)
22 V(I,J,K) = Q(I,J,K,3,M) / Q(I,J,K,1,M)
23 W(I,J,K) = Q(I,J,K,4,M) / Q(I,J,K,1,M)
24
25 KE = 0.5D+00 * (U(I,J,K) * U(I,J,K) +
26 > V(I,J,K) * V(I,J,K) +
27 > W(I,J,K) * W(I,J,K))
28
29 T(I,J,K) = (Q(I,J,K,5,M) / Q(I,J,K,1,M) - KE) / CVAIR
30 P(I,J,K) = Q(I,J,K,1,M) * RGAIR * T(I,J,K)
Info - Line 10
A loop starting at line 10 was scoped without errors.
A loop starting at line 10 is flat (contains no external calls).
A loop starting at line 10 was not vectorized because a better candidate was found at line 12.
A loop starting at line 11 is flat (contains no external calls).
leslie3d.pl loaded. xleslie3d_mpi+2601-70t loaded.


Reveal OpenMP Scoping
Scope Loops
Scoping Results
update.f: Loop@10
Name Type Scope Info
i Scalar Private
j Scalar Private
k Scalar Private
ke Scalar Private
dq Array Shared
icmax Scalar Shared
jcmax Scalar Shared
kcmax Scalar Shared
m Scalar Shared
n Scalar Shared
p Array Shared
q Array Shared
t Array Shared
u Array Shared
v Array Shared
w Array Shared
First/Last Private
Enable FirstPrivate
Enable LastPrivate
Reduction
None
Find Name:
Insert Directive
Show Directive
Close

Perftools-lite profiles

Original running on 8 nodes x 1 MPI

```
Table 1:  Profile by Function (limited entries shown)
  Samp% |     Samp | Imb. |  Imb. | Group
        |          | Samp | Samp% |  Function=[MAX10]
        |          |      |       |   PE=HIDE
 100.0% | 29,633.0 |   -- |    -- | Total
|-------------------------------------------------
|  98.5% | 29,183.5 |   -- |    -- | USER
||------------------------------------------------
||  27.1% |  8,030.6 | 43.4 |  0.6% | fluxj_
||  26.9% |  7,958.6 | 47.4 |  0.7% | fluxk_
||  22.8% |  6,745.0 | 79.0 |  1.3% | fluxi_
||   5.2% |  1,551.4 | 21.6 |  1.6% | extrapk_
||   5.1% |  1,499.5 | 18.5 |  1.4% | extrapi_
||   4.6% |  1,376.2 | 24.8 |  2.0% | update_
||   4.2% |  1,258.9 | 24.1 |  2.1% | extrapj_
||================================================
|   1.4% |    407.9 |   -- |    -- | MPI
|================================================
```

Threaded running on 8 nodes 1 MPIx44 threads

```
Table 1:  Profile by Function (limited entries shown)

  Samp% |     Samp | Imb. |  Imb. | Group
        |          | Samp | Samp% |  Function=[MAX10]
        |          |      |       |   PE=HIDE
        |          |      |       |    Thread=HIDE

 100.0% | 1,131.5 |   -- |    -- | Total
|-------------------------------------------------
|  75.1% |   849.9 |   -- |    -- | USER
||------------------------------------------------
||  14.9% |   169.0 |  9.0 |  5.8% | fluxk_.LOOP@li.33
||  10.4% |   117.4 | 14.6 | 12.7% | fluxj_.LOOP@li.33
||   9.4% |   106.4 | 13.6 | 13.0% | fluxi_.LOOP@li.25
||   5.8% |    65.6 | 17.4 | 23.9% | update_.LOOP@li.14
||   4.9% |    55.0 | 11.0 | 19.0% | extrapk_.LOOP@li.146
||   4.8% |    53.9 | 12.1 | 21.0% | mpicx_
||   3.5% |    40.1 |  2.9 |  7.6% | tmstep_
||   3.2% |    36.8 |  4.2 | 11.8% | extrapj_.LOOP@li.144
||================================================
|  16.3% |   184.0 |   -- |    -- | MPI
||------------------------------------------------
||   8.1% |    91.1 | 11.9 | 13.2% | MPI_REDUCE
||   4.9% |    55.5 | 15.5 | 24.9% | MPI_SEND
||================================================
|   8.5% |    95.6 |   -- |    -- | ETC
|================================================
```

Perftools-lite profiles

Original running on 8 nodes x 44 MPI

Table 1: Profile by Function (limited entries shown)

Samp%	Samp	Imb. Samp	Imb. Samp%	Group Function=[MAX10] PE=HIDE
100.0%	1,234.0	--	--	Total
66.0%	814.8	--	--	MPI
22.7%	280.2	141.8	33.7%	MPI_BARRIER
20.6%	254.3	215.7	46.0%	MPI_REDUCE
16.4%	201.8	113.2	36.0%	MPI_ALLREDUCE
4.8%	59.0	37.0	38.6%	MPI_SEND
30.5%	376.1	--	--	USER
6.0%	74.2	12.8	14.8%	fluxj_.LOOP@li.33
5.9%	73.4	18.6	20.3%	fluxi_.LOOP@li.25
5.7%	70.6	17.4	19.9%	fluxk_.LOOP@li.33
2.5%	31.4	11.6	27.0%	extrapk_.LOOP@li.146
2.5%	30.4	9.6	24.0%	extrapj_.LOOP@li.144
3.4%	41.7	--	--	ETC
2.7%	33.1	5.9	15.2%	__cray_dset_HSW

Threaded running on 8 nodes 1 MPIx44 threads

Table 1: Profile by Function (limited entries shown)

Samp%	Samp	Imb. Samp	Imb. Samp%	Group Function=[MAX10] PE=HIDE Thread=HIDE
100.0%	1,131.5	--	--	Total
75.1%	849.9	--	--	USER
14.9%	169.0	9.0	5.8%	fluxk_.LOOP@li.33
10.4%	117.4	14.6	12.7%	fluxj_.LOOP@li.33
9.4%	106.4	13.6	13.0%	fluxi_.LOOP@li.25
5.8%	65.6	17.4	23.9%	update_.LOOP@li.14
4.9%	55.0	11.0	19.0%	extrapk_.LOOP@li.146
4.8%	53.9	12.1	21.0%	mpicx_
3.5%	40.1	2.9	7.6%	tmstep_
3.2%	36.8	4.2	11.8%	extrapj_.LOOP@li.144
16.3%	184.0	--	--	MPI
8.1%	91.1	11.9	13.2%	MPI_REDUCE
4.9%	55.5	15.5	24.9%	MPI_SEND
8.5%	95.6	--	--	ETC

Why is OpenMP doing so poor?

- OpenMP run
  - setenv OMP_NUM_THREADS 44
  - aprun -n 8 -N 1 -d 44 ./xleslie3d_mpi > out8on8_44
- MPI run
  - setenv OMP_NUM_THREADS 1
  - aprun -n 352 -N 44 -S 22 ./xleslie3d_mpi > all_mpi

Why are the OpenMP loops doing poorly?

Table 3: Memory Bandwidth by Numanode (limited entries shown)

Memory Traffic GBytes	Local Memory Traffic GBytes	Remote Memory Traffic GBytes	Thread Time	Memory Traffic GBytes / Sec	Memory Traffic / Nominal Peak	Numanode Node Id=[max3,min3] PE=HIDE Thread=HIDE
184.47	173.59	10.89	11.578777	15.93	20.7%	numanode.0
183.50	173.59	9.91	11.569322	15.86	20.7%	nid.63
182.61	172.40	10.21	11.578777	15.77	20.5%	nid.61
178.55	167.75	10.80	11.563156	15.44	20.1%	nid.71
178.10	168.14	9.96	11.562097	15.40	20.1%	nid.62
178.08	168.07	10.01	11.564512	15.40	20.1%	nid.68
178.01	167.20	10.82	11.572032	15.38	20.0%	nid.70
60.36	14.73	45.62	9.073119	6.65	8.7%	numanode.1
60.36	14.73	45.62	9.072693	6.65	8.7%	nid.63
59.88	14.33	45.55	9.071553	6.60	8.6%	nid.62
59.48	14.19	45.29	9.068044	6.56	8.5%	nid.68
58.78	13.70	45.08	9.069259	6.48	8.4%	nid.70
58.67	13.87	44.81	9.071591	6.47	8.4%	nid.69
58.53	13.86	44.67	9.067146	6.46	8.4%	nid.71

This is the NUMA traffic from the all-MPI run.

Table 3: Memory Bandwidth by Numanode (limited entries shown)

Memory Traffic GBytes	Local Memory Traffic GBytes	Remote Memory Traffic GBytes	Thread Time	Memory Traffic GBytes / Sec	Memory Traffic / Nominal Peak	Numanode Node Id=[max3,min3] PE=HIDE
172.95	171.48	1.48	19.755654	8.75	11.4%	numanode.0
172.77	171.48	1.30	19.414237	8.90	11.6%	nid.68
172.09	170.61	1.48	19.071340	9.02	11.7%	nid.63
171.20	169.93	1.27	17.631761	9.71	12.6%	nid.62
162.51	161.07	1.43	19.675857	8.26	10.8%	nid.71
162.28	160.82	1.46	19.730793	8.22	10.7%	nid.72
161.75	160.29	1.46	19.755654	8.19	10.7%	nid.70
168.69	166.81	1.89	19.781479	8.53	11.1%	numanode.1
168.69	166.81	1.89	19.454144	8.67	11.3%	nid.62
167.74	166.03	1.71	19.476164	8.61	11.2%	nid.63
166.66	164.88	1.78	19.225409	8.67	11.3%	nid.61
161.68	160.07	1.61	19.781479	8.17	10.6%	nid.71
161.60	159.99	1.62	19.642791	8.23	10.7%	nid.70
157.32	156.01	1.31	18.036118	8.72	11.4%	nid.72

Perftools-lite profiles

Original running on 8 nodes x 44 MPI

Table 1: Profile by Function (limited entries shown)

Samp%	Samp	Imb. Samp	Imb. Samp%	Group Function=[MAX10] PE=HIDE
100.0%	1,234.0	--	--	Total
66.0%	814.8	--	--	MPI
22.7%	280.2	141.8	33.7%	MPI_BARRIER
20.6%	254.3	215.7	46.0%	MPI_REDUCE
16.4%	201.8	113.2	36.0%	MPI_ALLREDUCE
4.8%	59.0	37.0	38.6%	MPI_SEND
30.5%	376.1	--	--	USER
6.0%	74.2	12.8	14.8%	fluxj_.LOOP@li.33
5.9%	73.4	18.6	20.3%	fluxi_.LOOP@li.25
5.7%	70.6	17.4	19.9%	fluxk_.LOOP@li.33
2.5%	31.4	11.6	27.0%	extrapk_.LOOP@li.146
2.5%	30.4	9.6	24.0%	extrapj_.LOOP@li.144
3.4%	41.7	--	--	ETC
2.7%	33.1	5.9	15.2%	__cray_dset_HSW

Threaded running on 8 nodes 2 MPIx22 threads

Ahh – Much Better

Table 1: Profile by Function (limited entries shown)

Samp%	Samp	Imb. Samp	Imb. Samp%	Group Function=[MAX10] PE=HIDE Thread=HIDE
100.0%	843.8	--	--	Total
55.0%	463.9	--	--	USER
9.9%	83.1	9.9	11.3%	fluxk_.LOOP@li.33
9.2%	77.6	8.4	10.5%	fluxj_.LOOP@li.33
8.7%	73.3	14.7	17.8%	fluxi_.LOOP@li.25
3.6%	30.1	8.9	24.3%	extrapk_.LOOP@li.146
3.5%	29.8	10.2	27.3%	mpicx_
3.4%	28.4	6.6	20.0%	update_.LOOP@li.14
37.4%	315.4	--	--	MPI
19.4%	164.0	51.0	25.3%	MPI_REDUCE
9.4%	79.1	36.9	33.9%	MPI_ALLREDUCE
6.4%	54.1	27.9	36.3%	MPI_SEND
7.5%	63.2	--	--	ETC
2.9%	24.2	0.8	3.5%	__cray_dset_HSW

Lets look at perftools-lite-hbm

```
Table 5:  Profile by Group, Function, and Line (limited entries shown)

  Samp% | Samp | Imb. |  Imb. | MEM_LOAD_UOPS_RETIRED | RESOURCE_STALLS | Group
        |      | Samp | Samp% |    :HIT_LFB:precise=2 |            :ANY |  Function=[MAX10]
        |      |      |       |                       |                  |   Source
        |      |      |       |                       |                  |    Line
        |      |      |       |                       |                  |     PE=HIDE
 100.0% | 59.5 |   -- |    -- | 5,805,421,397,144,252 | 209,611,978,880 | Total
|-------------------------------------------------------------------------------
|  97.1% | 57.8 |   -- |    -- | 5,594,315,164,526,714 | 208,307,562,822 | USER
||------------------------------------------------------------------------------
||  23.5% | 14.0 |   -- |    -- | 1,231,453,023,644,516 |  56,564,956,788 | fluxj_
3|       |      |      |       |                       |                 |  fluxj.f
||||-----------------------------------------------------------------------------
4|||   4.2% |  2.5 |  1.5 | 42.9% |   246,290,604,776,946 |  11,331,973,206 | line.36
4|||   7.8% |  4.6 |  5.4 | 61.4% |   387,028,093,089,632 |  18,250,631,160 | line.72
4|||   8.0% |  4.8 |  4.2 | 54.0% |   316,659,349,107,561 |  16,880,190,381 | line.81
||||=============================================================================
||  23.3% | 13.9 |   -- |    -- | 1,618,481,116,517,358 |  46,077,159,986 | fluxi_
3|       |      |      |       |                       |                 |  fluxi.f
||||-----------------------------------------------------------------------------
4|||   5.7% |  3.4 |  1.6 | 37.1% |   316,659,348,989,901 |   9,236,821,683 | line.29
4|||   1.5% |  0.9 |  2.1 | 81.0% |    70,368,744,186,054 |   5,096,471,788 | line.41
4|||   1.1% |  0.6 |  1.4 | 78.6% |    70,368,744,181,663 |   2,885,621,509 | line.42
4|||   1.7% |  1.0 |  1.0 | 57.1% |   105,553,116,300,240 |   5,639,566,321 | line.48
4|||   7.8% |  4.6 |  3.4 | 48.2% |   668,503,069,752,688 |  13,084,700,740 | line.61
4|||   4.8% |  2.9 |  1.1 | 32.1% |   316,659,348,925,684 |   7,922,852,884 | line.68
```

Lets look at perftools-lite-hbm

```
28.                          ! Directive inserted by Cray Reveal.  May be incomplete.
29.    M---------------------< !$OMP  parallel do default(none)
30.    M                     !$OMP&   private (fsj,i,j,k,l,qs,qsp,qspj)
31.    M                     !$OMP&   shared  (dq,dtv,ind,jadd,jcmax,kcmax,pav,qav,sjx,sjy,sjz,u,uav,
32.    M                     !$OMP&            v,vav,w,wav)
33.  + M mF------------------<       DO K = 1, KCMAX
34.  + M mF F---------------<          DO J = 0, JCMAX
35.    M mF F
36.  + M mF F fVF----------<>             QS(1:IND) = UAV(1:IND,J,K) * SJX(1:IND,J,K) +
37.    M mF F                      >                 VAV(1:IND,J,K) * SJY(1:IND,J,K) +
38.    M mF F                      >                 WAV(1:IND,J,K) * SJZ(1:IND,J,K)
39.    M mF F
40.    M mF F                           IF (NSCHEME .EQ. 2) THEN
41.    M mF F                              L = J + 1 - JADD
42.    M mF F D-------------<              DO I = 1, IND
43.    M mF F D                               QSP = U(I,L,K) * SJX(I,J,K) +
44.    M mF F D                    >                V(I,L,K) * SJY(I,J,K) +
45.    M mF F D                    >                W(I,L,K) * SJZ(I,J,K)
46.    M mF F D                               QSPJ = (QSP - QS(I)) * DBLE(1 - 2 * JADD)
47.    M mF F D                               IF (QSPJ .GT. 0.0D+00) QS(I) = 0.5D+00 * (QS(I) + QSP)
48.    M mF F D------------->              ENDDO
49.    M mF F                           ENDIF
50.    M mF F
51.  + M mF F fF----------<>           FSJ(1:IND,J,1) =  QAV(1:IND,J,K,1) * QS(1:IND)
52.  + M mF F fF----------<>           FSJ(1:IND,J,2) =  QAV(1:IND,J,K,2) * QS(1:IND) +
53.    M mF F                      >                      PAV(1:IND,J,K) * SJX(1:IND,J,K)
54.  + M mF F fF----------<>           FSJ(1:IND,J,3) =  QAV(1:IND,J,K,3) * QS(1:IND) +
55.    M mF F                      >                      PAV(1:IND,J,K) * SJY(1:IND,J,K)
```

Lets look at perftools-lite-hbm

```
73.     M mF
74.   + M mF ib--------------<         DO J = 1, JCMAX
75.   + M mF ib ib-----------<            DO L = 1, 5
76.     M mF ib ib Vbr2-----<>               DQ(1:IND,J,K,L) = DQ(1:IND,J,K,L) -
77.     M mF ib ib                  >            DTV(1:IND,J,K) * (FSJ(1:IND,J,L) - FSJ(1:IND,J-1,L))
78.     M mF ib ib----------->            ENDDO
79.     M mF ib
80.     M mF ib                           IF (ISGSK .EQ. 1) THEN
81.     M mF ib                              DQ(1:IND,J,K,7) = DQ(1:IND,J,K,7) -
82.     M mF ib                     >            DTV(1:IND,J,K) * (FSJ(1:IND,J,7) - FSJ(1:IND,J-1,7))
83.     M mF ib                           ENDIF
84.     M mF ib
85.     M mF ib                           IF ( ICHEM .GT. 0 ) THEN
86.     M mF ib D-----------<                DO L = 8, 7 + NSPECI
87.     M mF ib D                               DQ(1:IND,J,K,L) = DQ(1:IND,J,K,L) -
88.     M mF ib D                   >             DTV(1:IND,J,K) * (FSJ(1:IND,J,L) - FSJ(1:IND,J-1,L))
89.     M mF ib D----------->                ENDDO
90.     M mF ib                           ENDIF
91.     M mF ib------------->          ENDDO
92.     M mF---------------->>       ENDDO
```

Perftools-lite profiles

After removing array syntax

Table 1: Profile by Function (limited entries shown)

Samp%	Samp	Imb. Samp	Imb. Samp%	Group Function=[MAX10] PE=HIDE Thread=HIDE
100.0%	803.9	--	--	Total
54.0%	433.8	--	--	USER
9.7%	78.0	6.0	7.6%	fluxk_.LOOP@li.32
8.5%	68.3	7.7	10.8%	fluxj_.LOOP@li.32
8.1%	65.1	5.9	8.8%	fluxi_.LOOP@li.25
3.5%	28.4	3.6	12.1%	extrapk_.LOOP@li.150
3.3%	26.8	12.2	33.5%	mpicx_
3.3%	26.6	2.4	9.0%	update_.LOOP@li.14
37.7%	302.8	--	--	MPI
22.0%	176.8	56.2	25.8%	MPI_REDUCE
10.2%	81.7	46.3	38.6%	MPI_ALLREDUCE
3.9%	31.2	14.8	34.2%	MPI_SEND
8.2%	66.0	--	--	ETC
3.1%	25.3	1.7	6.7%	__cray_dset_HSW

Table 1: Profile by Function (limited entries shown)

Samp%	Samp	Imb. Samp	Imb. Samp%	Group Function=[MAX10] PE=HIDE Thread=HIDE
100.0%	843.8	--	--	Total
55.0%	463.9	--	--	USER
9.9%	83.1	9.9	11.3%	fluxk_.LOOP@li.33
9.2%	77.6	8.4	10.5%	fluxj_.LOOP@li.33
8.7%	73.3	14.7	17.8%	fluxi_.LOOP@li.25
3.6%	30.1	8.9	24.3%	extrapk_.LOOP@li.146
3.5%	29.8	10.2	27.3%	mpicx_
3.4%	28.4	6.6	20.0%	update_.LOOP@li.14
37.4%	315.4	--	--	MPI
19.4%	164.0	51.0	25.3%	MPI_REDUCE
9.4%	79.1	36.9	33.9%	MPI_ALLREDUCE
6.4%	54.1	27.9	36.3%	MPI_SEND
7.5%	63.2	--	--	ETC
2.9%	24.2	0.8	3.5%	__cray_dset_HSW

So lets add OpenACC directives

```
#ifdef _OPENACC
!$ACC PARALLEL LOOP PRIVATE (fsi,i,j,k,l,qs,qsp,qspi)
#else
! Directive inserted by Cray Reveal.  May be incomplete.
!$OMP  parallel do default(none)
!$OMP&   private (fsi,i,j,k,l,qs,qsp,qspi)
!$OMP&   shared  (dq,dtv,iadd,icmax,jcmax,kcmax,pav,qav,six,siy,siz,u,
!$OMP&            uav,v,vav,w,wav)
#endif
      DO K = 1, KCMAX
         DO J = 1, JCMAX
          DO I = 0, ICMAX
            QS(I) = UAV(I,J,K) * SIX(I,J,K) +
     >                VAV(I,J,K) * SIY(I,J,K) +
     >                WAV(I,J,K) * SIZ(I,J,K)


            IF ( NSCHEME .EQ. 2 ) THEN
                  L = I + 1 - IADD
                  QSP = U(L,J,K) * SIX(I,J,K) +
     >                  V(L,J,K) * SIY(I,J,K) +
     >                  W(L,J,K) * SIZ(I,J,K)
                  QSPI = (QSP - QS(I)) * DBLE(1 - 2 * IADD)
                  IF ( QSPI .GT. 0.0D0 ) QS(I) = 0.5D0 * (QS(I) + QSP)
            ENDIF
```

Only need to worry about private variables

Don’t worry about data movement yet

So lets add OpenACC directives

```
21.                    #ifdef _OPENACC
22.  + G--------------< !$ACC PARALLEL LOOP PRIVATE (fsi,i,j,k,l,qs,qsp,qspi)
23.    G               #else
24.    D               ! Directive inserted by Cray Reveal.  May be incomplete.
25.    D               !$OMP  parallel do default(none)
26.    D               !$OMP&   private (fsi,i,j,k,l,qs,qsp,qspi)
27.    D               !$OMP&   shared  (dq,dtv,iadd,icmax,jcmax,kcmax,pav,qav,six,siy,siz,u,
28.    D               !$OMP&            uav,v,vav,w,wav)
29.    G               #endif
30.  + G gF-----------<       DO K = 1, KCMAX
31.  + G gF F---------<          DO J = 1, JCMAX
32.    G gF F g-------<           DO I = 0, ICMAX
33.    G gF F g                    QS(I) = UAV(I,J,K) * SIX(I,J,K) +
34.    G gF F g             >                VAV(I,J,K) * SIY(I,J,K) +
35.    G gF F g             >                WAV(I,J,K) * SIZ(I,J,K)
36.    G gF F g
37.    G gF F g                    IF ( NSCHEME .EQ. 2 ) THEN
38.    G gF F g                         L = I + 1 - IADD
39.    G gF F g                         QSP = U(L,J,K) * SIX(I,J,K) +
40.    G gF F g             >                V(L,J,K) * SIY(I,J,K) +
41.    G gF F g             >                W(L,J,K) * SIZ(I,J,K)
42.    G gF F g                         QSPI = (QSP - QS(I)) * DBLE(1 - 2 * IADD)
43.    G gF F g                         IF ( QSPI .GT. 0.0D0 ) QS(I) = 0.5D0 * (QS(I) + QSP)
44.    G gF F g                    ENDIF
45.    G gF F g                    FSI(I,1) =  QAV(I,J,K,1) * QS(I)
46.    G gF F g                    FSI(I,2) =  QAV(I,J,K,2) * QS(I) +
47.    G gF F g             >                           PAV(I,J,K) * SIX(I,J,K)
```

So lets add OpenACC directives

Let the compiler figure out data movement

```
ftn-6405 ftn: ACCEL FLUXI, File = fluxi.f, Line = 22
  A region starting at line 22 and ending at line 88 was placed on the accelerator.

ftn-6418 ftn: ACCEL FLUXI, File = fluxi.f, Line = 22
  If not already present: allocate memory and copy whole array "siy" to accelerator, free at line 88 (acc_copyin).

ftn-6418 ftn: ACCEL FLUXI, File = fluxi.f, Line = 22
  If not already present: allocate memory and copy whole array "vav" to accelerator, free at line 88 (acc_copyin).

ftn-6418 ftn: ACCEL FLUXI, File = fluxi.f, Line = 22
  If not already present: allocate memory and copy whole array "six" to accelerator, free at line 88 (acc_copyin).

ftn-6418 ftn: ACCEL FLUXI, File = fluxi.f, Line = 22
  If not already present: allocate memory and copy whole array "uav" to accelerator, free at line 88 (acc_copyin).

ftn-6418 ftn: ACCEL FLUXI, File = fluxi.f, Line = 22
  If not already present: allocate memory and copy whole array "siz" to accelerator, free at line 88 (acc_copyin).

ftn-6418 ftn: ACCEL FLUXI, File = fluxi.f, Line = 22
  If not already present: allocate memory and copy whole array "wav" to accelerator, free at line 88 (acc_copyin).
```

Approach for turning OpenMP threading into OpenACC

- First convert all OpenMP high level loops into OpenACC
- Next introduce DATA regions up the call tree
- Will also use perfttools to investigate accelerator performance
  - Module load perftools instead of perftools-lite
  - After building application pat_build –u –g mpi <executable>
  - Run <executable+pat>
  - Pat_report < directory created when running <executable+pat>

Perftools-lite loops – Run on 8 nodes–8 MPI tasks

Can we introduce DATA regions here?

```
Table 1:  Function Calltree View (limited entries shown)
  Time% |       Time |       Calls | Calltree
        |            |             |  PE=HIDE
 100.0% | 331.356157 |          -- | Total
|-------------------------------------------------------------
| 100.0% | 331.356117 |         2.0 | les3d_
|  99.3% | 329.201866 |          -- |  les3d_.LOOP.3.li.216
3  98.7% | 327.029239 |          -- |   les3d_.LOOP.4.li.272
||||-----------------------------------------------------------
4|||  31.6% | 104.588754 |     1,000.0 | fluxk_
|||||-----------------------------------------------------------
5||||  20.2% |  66.813949 |          -- | fluxk_.LOOP.1.li.28
6||||       |            |              |  fluxk_.LOOP.2.li.29
7||||  20.2% |  66.813949 | 9,312,000.0 |   visck_
5||||   6.4% |  21.068680 |     1,000.0 | fluxk_(exclusive)
5||||   5.0% |  16.706125 |     1,000.0 | extrapk_
|||||===========================================================
4|||  29.2% |  96.783424 |     1,000.0 | fluxi_
|||||-----------------------------------------------------------
5||||  16.9% |  55.929620 |          -- | fluxi_.LOOP.1.li.21
6||||       |            |              |  fluxi_.LOOP.2.li.22
7||||  16.9% |  55.929620 | 9,216,000.0 |   visci_
5||||   6.8% |  22.465390 |     1,000.0 | extrapi_
|||||-----------------------------------------------------------
6|||||   3.4% |  11.399343 |     1,000.0 | extrapi_(exclusive)
6|||||   3.3% |  11.066047 |          -- | extrapi_.LOOP.1.li.128
7|||||       |            |              |  extrapi_.LOOP.2.li.129
8|||||   3.3% |  11.066047 | 9,801,000.0 |   exi4_
|||||===========================================================
5||||   5.5% |  18.388414 |     1,000.0 | fluxi_(exclusive)
|||||===========================================================
4|||  28.4% |  93.988073 |     1,000.0 | fluxj_
|||||-----------------------------------------------------------
5||||  18.3% |  60.687484 |          -- | fluxj_.LOOP.1.li.28
6||||       |            |              |  fluxj_.LOOP.2.li.29
7||||  18.3% |  60.687484 | 9,312,000.0 |   viscj_
5||||   5.9% |  19.520634 |     1,000.0 | fluxj_(exclusive)
5||||   4.2% |  13.779955 |     1,000.0 | extrapj_
|||||===========================================================
4|||   5.3% |  17.668610 |     1,000.0 | parallel_
|||||-----------------------------------------------------------
5||||   4.0% |  13.109379 |          -- | parallel_.LOOP.1.li.16
6||||   4.0% |  13.109379 |     5,000.0 |  mpicx_
5||||   1.4% |   4.543585 |     4,000.0 | ghost_
|||||===========================================================
```

This is the timestep loop in main

```
215.                    #ifdef _OPENACC
216.  +                 !$ACC DATA COPY(P,Q,SIY,VAV,SIX,UAV,SIZ,WAV,QAV,PAV,U,V,W,T,Y,W,T11,T21,
217.                    !$ACC&    EKAV,EK,DS,DS1,T31,TAV,T12,T22,T32,T13,T23,T33,HF,DTV,DQ)
218.                    #endif
219.  + 1-------------<       DO WHILE ( NADV .LE. NEND .AND. TAU .LT. 35.0D+00 )
220.    1
221.    1                       IF ( MOD ( NADV, ITIME ) .EQ. 0 .OR. NADV .EQ. NSTART )
222.  + 1                    >                              CALL TMSTEP ( TMAX, RMACH )
223.    1
224.    1                       TIME = TIME + DT
225.    1
226.  + 1                       CALL ANALYSIS ( TAU, DELM )
227.    1
228.    1                       IF ( IAM .EQ. 0 .AND. MOD ( NADV, ITIME ) .EQ. 0 ) THEN
229.    1                          WRITE(6,1000) NADV, DT, TIME, TAU, DELM
230.    1                          WRITE(50,'(2(F10.5,1X))') TAU, DELM
231.    1                       ENDIF
232.    1                 1000  FORMAT(' NADV = ', I6,
233.    1                    >         ' DT   = ', E10.4,
234.    1                    >         ' TIME = ', F8.5,
235.    1                    >         ' TAU  = ', F8.4,
236.    1                    >         ' DELM = ', F8.4)
237.    1
238.    1
239.    1                       IF ( IDYN .GT. 0 ) THEN
240.  + 1                          CALL EJECT ( 'LDKM, PLEASE RECOMPILE WITH -DLDKM!' )
241.    1                       ENDIF
242.    1
```

But now we have more work to do

- Must either put all the work within the DATA region on the accelerator, or
- Transfer data back and forth to the host

Additional computation and boundary exchange

```
#ifdef _OPENACC
!$ACC PARALLEL LOOP PRIVATE(L)
#else
!$OMP PARALLEL DO PRIVATE(L)
#endif
              DO K = 1, KCMAX
                 DO J = 1, JCMAX
                    Q(I,J,K,1,M) = BOUND(J,K,1)
                    Q(I,J,K,2,M) = BOUND(J,K,1) * BOUND(J,K,2)
                    Q(I,J,K,3,M) = BOUND(J,K,1) * BOUND(J,K,3)
                    Q(I,J,K,4,M) = BOUND(J,K,1) * BOUND(J,K,4)
                    T(I,J,K)     = BOUND(J,K,5)
                    IF ( ICHEM .GT. 0 ) THEN
                       DO NS = 1, NSPECI
                          L = 7 + NS
                          Q(I,J,K,L,M) = BOUND(J,K,1) * BOUND(J,K,L)
                       ENDDO
                    ENDIF
                    IF ( ISGSK .EQ. 1 ) THEN
                       Q(I,J,K,7,M) = BOUND(J,K,1) * BOUND(J,K,7)
                    ENDIF
                 ENDDO
              ENDDO
           ENDIF
```

```
! Exchange boundary information
#ifdef _OPENACC
!$ACC UPDATE HOST(Q,HF)
#endif
            IF ( NSCHEME .EQ. 2 ) THEN
!               CALL PARALLEL(2000 + N, 1)
               CALL PARALLEL(2000 + N)
            ELSE
!               CALL PARALLEL(2000 + N, 2)
               CALL PARALLEL(2000 + N)
            ENDIF
#ifdef _OPENACC
!$ACC UPDATE DEVICE(Q,U,V,W)
```

Using Perftools to look at GPU performance

```
Table 1:  Profile by Function Group and Function (limited entries shown)
  Time% |       Time |      Imb. |  Imb. |     Calls | Group
        |            |      Time | Time% |           |  Function
        |            |           |       |           |   PE=HIDE
        |            |           |       |           |    Thread=HIDE
 100.0% | 49.872536 |        -- |    -- | 17,417.0 | Total
|-------------------------------------------------------------------------
|  85.1% | 42.464541 |        -- |    -- |  8,203.0 | OACC
||------------------------------------------------------------------------
||  26.1% | 13.028909 | 0.018355 |  0.3% |    200.0 | les3d_.ACC_COPY@li.353
||  20.0% |  9.995882 | 0.002718 |  0.1% |    200.0 | les3d_.ACC_COPY@li.343
||   7.6% |  3.790519 | 0.002323 |  0.1% |    200.0 | fluxi_.ACC_SYNC_WAIT@li.88
||   7.2% |  3.582807 | 0.000122 |  0.0% |    200.0 | fluxj_.ACC_SYNC_WAIT@li.103
||   6.7% |  3.340761 | 0.000214 |  0.0% |    200.0 | fluxk_.ACC_SYNC_WAIT@li.100
||   4.3% |  2.165933 | 0.003824 |  0.4% |    200.0 | fluxj_.ACC_COPY@li.30
||   4.3% |  2.153276 | 0.003472 |  0.3% |    200.0 | fluxk_.ACC_COPY@li.29
||   1.9% |  0.967199 | 0.001804 |  0.4% |    200.0 | extrapj_.ACC_SYNC_WAIT@li.229
||   1.7% |  0.843420 | 0.009258 |  2.2% |    200.0 | extrapk_.ACC_SYNC_WAIT@li.230
||   1.4% |  0.688849 | 0.013444 |  3.8% |      1.0 | les3d_.ACC_COPY@li.216
||   1.1% |  0.572288 | 0.005227 |  1.8% |      1.0 | les3d_.ACC_COPY@li.413
||   1.0% |  0.514717 | 0.000140 |  0.1% |    200.0 | les3d_.ACC_SYNC_WAIT@li.343
||========================================================================
|  12.6% |  6.269397 |        -- |    -- |  5,949.0 | USER
||------------------------------------------------------------------------
||   2.4% |  1.208565 | 0.004756 |  0.8% |      1.0 | grid_
||   2.2% |  1.099410 | 0.002228 |  0.4% |     21.0 | tmstep_
||   2.0% |  0.995479 | 0.000204 |  0.0% |    606.0 | ghost_
||   1.6% |  0.776549 | 0.011156 |  2.8% |      1.0 | setiv_
||   1.2% |  0.611475 | 0.001712 |  0.6% |  1,010.0 | mpicx_
||   1.1% |  0.557527 | 0.002275 |  0.8% |    603.0 | wallbc_
```

But what is the future of OpenACC?

- Newer accelerator installations are using OpenMP directives for accelerators
  - As OpenMP compilers become more robust, OpenACC could easily become extinct.

So we convert to OpenMP 4.5

```
13.                    !$omp declare target (visci)
14.
15.                    !     ALLOCATE ( QS(0:ICMAX), FSI(0:ICMAX,ND), STAT = ERROR )
16.                    !     IF ( ERROR .NE. 0 ) CALL EJECT ('ALLOCATION ERROR: FLUXI')
17.  +                 !$omp target data map(tofrom:SIY,VAV,SIX,UAV,SIZ,WAV,QAV,PAV,U,V,W,T,
18.                    !$omp&         Y,W,T11,T21,Q,HF,DS,DS1,T31,TAV,T12,T22,T32,T13,T23,
19.                    !$omp&         T33,HF,DTV,DQ)
20.  +                       CALL EXTRAPI ( FSI )
21.  + G--------------< !$omp target teams distribute
22.    G                !$omp&           private(fsi,i,j,k,l,qs,qsp,qspi)
23.  + G gF-----------<       DO K = 1, KCMAX
24.  + G gF F---------<          DO J = 1, JCMAX
25.    G gF F g-------<           DO I = 0, ICMAX
26.    G gF F g                    QS(I) = UAV(I,J,K) * SIX(I,J,K) +
27.    G gF F g             >                  VAV(I,J,K) * SIY(I,J,K) +
28.    G gF F g             >                  WAV(I,J,K) * SIZ(I,J,K)
29.    G gF F g
30.    G gF F g                    IF ( NSCHEME .EQ. 2 ) THEN
31.    G gF F g                         L = I + 1 - IADD
32.    G gF F g                         QSP = U(L,J,K) * SIX(I,J,K) +
33.    G gF F g             >                V(L,J,K) * SIY(I,J,K) +
34.    G gF F g             >                W(L,J,K) * SIZ(I,J,K)
35.    G gF F g                         QSPI = (QSP - QS(I)) * DBLE(1 - 2 * IADD)
36.    G gF F g                         IF ( QSPI .GT. 0.0D0 ) QS(I) = 0.5D0 * (QS(I) + QSP)
37.    G gF F g                    ENDIF
38.    G gF F g                    FSI(I,1) =  QAV(I,J,K,1) * QS(I)
39.    G gF F g                    FSI(I,2) =  QAV(I,J,K,2) * QS(I) +
40.    G gF F g             >                           PAV(I,J,K) * SIX(I,J,K)
```

So we convert to OpenMP 4.5

```
216.  +              !$omp target data map(tofrom:P,Q,SIY,VAV,SIX,UAV,SIZ,WAV,QAV,PAV,U,V,
217.                 !$omp&        W,T,Y,W,T11,T21,EKAV,EK,DS,DS1,T31,TAV,T12,T22,T32,T13,
218.                 !$omp&        T23,T33,HF,DTV,DQ)
220.  + 1----------<       DO WHILE ( NADV .LE. NEND .AND. TAU .LT. 35.0D+00 )
221.    1
222.    1                    IF ( MOD ( NADV, ITIME ) .EQ. 0 .OR. NADV .EQ. NSTART )
223.  + 1                 >                               CALL TMSTEP ( TMAX, RMACH )
224.    1
225.    1                    TIME = TIME + DT
226.    1
227.  + 1                    CALL ANALYSIS ( TAU, DELM )
228.    1
229.    1                    IF ( IAM .EQ. 0 .AND. MOD ( NADV, ITIME ) .EQ. 0 ) THEN
230.    1                       WRITE(6,1000) NADV, DT, TIME, TAU, DELM
231.    1                       WRITE(50,'(2(F10.5,1X))') TAU, DELM
232.    1                    ENDIF
233.    1              1000  FORMAT(' NADV = ', I6,
234.    1                   >        ' DT   = ', E10.4,
235.    1                   >        ' TIME = ', F8.5,
236.    1                   >        ' TAU  = ', F8.4,
237.    1                   >        ' DELM = ', F8.4)
238.    1
239.    1
240.    1                    IF ( IDYN .GT. 0 ) THEN
241.  + 1                       CALL EJECT ( 'LDKM, PLEASE RECOMPILE WITH -DLDKM!' )
242.    1                    ENDIF
243.    1
244.    1                    IADD = MOD ( NADV, 2 )
```

So we convert to OpenMP 4.5

```
344.     1 2
345.     1 2          ! Exchange boundary information
347.     1 2          !$omp target update from(q,hf)
349.     1 2                    IF ( NSCHEME .EQ. 2 ) THEN
350.     1 2          !              CALL PARALLEL(2000 + N, 1)
351.   + 1 2                       CALL PARALLEL(2000 + N)
352.     1 2                    ELSE
353.     1 2          !              CALL PARALLEL(2000 + N, 2)
354.   + 1 2                       CALL PARALLEL(2000 + N)
355.     1 2                    ENDIF
356.     1 2          #ifdef _OPENACC
357.     1 2          !$omp target update to(q,u,v,w)
358.     1 2          #endif
359.     1 2
```

Really more to do

- We are updating the entire array for the halo exchange
  - Need to go into PARALLEL and put packing/unpacking of halos on the accelerator and use GPU Direct to transfer halos between accelerators

- Also we can select to run on the accelerator or using threading on the host by choosing which module to load

  - craype-accel-nvidia60. To run on accelerator

  - craype-accel-host. To run threaded on the host


Questions?
CRAY