On the Role of Directives in High-Performance Computing

Doug Miles
PGI Compilers & Tools
NVIDIA Corporation
CCDSC, October 2016

PGI®

CDIR$ IVDEP

```
CMIC$ DOALL
```

OpenACC KERNELS as a construct …

```
% pgfortran –fast –acc –Minfo -c PdV_kernel.f90
pdv_kernel:
    ...
    77, Loop is parallelizable
    79, Loop is parallelizable
        Accelerator kernel generated
        Generating Tesla code
        77, !$acc loop gang, vector(4) ! blockidx%y
                                       ! threadidx%y
        79, !$acc loop gang, vector(32)! blockidx%x
                                       ! threadidx%x
    ...
```

```
% pgfortran –fast –ta=multicore ... PdV_kernel.f90
pdv_kernel:
    77, Loop is parallelizable
         Generating Multicore code
         77, !$acc loop gang
    79, Loop is parallelizable
         3 loop-carried redundant expressions removed
           with 9 operations and 9 arrays
         Innermost loop distributed: 2 new loops
         Generated vector sse code for the loop
         Generated 2 prefetch instructions for the loop
         Generated 12 prefetch instructions for the loop
...
```

```
75 !$ACC KERNELS
76 !$ACC LOOP INDEPENDENT
77     DO k=y_min,y_max
78 !$ACC LOOP INDEPENDENT PRIVATE(right_flux,left_flux,top_flux,bottom_flux,total_flux,
                              min_cell_volume,energy_change,recip_volume)
79       DO j=x_min,x_max
80
81         left_flux=  (xarea(j  ,k  )*(xvel0(j  ,k  )+xvel0(j  ,k+1)                &
82                                     +xvel0(j  ,k  )+xvel0(j  ,k+1)))*0.25_8*dt*0.5
83         right_flux= (xarea(j+1,k  )*(xvel0(j+1,k  )+xvel0(j+1,k+1)                &
84                                     +xvel0(j+1,k  )+xvel0(j+1,k+1)))*0.25_8*dt*0.5
85         bottom_flux=(yarea(j  ,k  )*(yvel0(j  ,k  )+yvel0(j+1,k  )                &
86                                     +yvel0(j  ,k  )+yvel0(j+1,k  )))*0.25_8*dt*0.5
87         top_flux=   (yarea(j  ,k+1)*(yvel0(j  ,k+1)+yvel0(j+1,k+1)                &
88                                     +yvel0(j  ,k+1)+yvel0(j+1,k+1)))*0.25_8*dt*0.5
89         total_flux=right_flux-left_flux+top_flux-bottom_flux
90
91         volume_change(j,k)=volume(j,k)/(volume(j,k)+total_flux)
92
93         min_cell_volume=MIN(volume(j,k)+right_flux-left_flux+top_flux-bottom_flux &
94                            ,volume(j,k)+right_flux-left_flux                      &
95                            ,volume(j,k)+top_flux-bottom_flux)
97         recip_volume=1.0/volume(j,k)
99         energy_change=(pressure(j,k)/density0(j,k)+viscosity(j,k)/density0(j,k))*...
101        energy1(j,k)=energy0(j,k)-energy_change
103        density1(j,k)=density0(j,k)*volume_change(j,k)
105      ENDDO
106    ENDDO
107 !$ACC END KERNELS
```

PGI®

… and a path to standard parallel languages

Fortran 2008/2015 DO CONCURRENT

+ True Parallel Loops
+ Loop-scope shared/private data
− No support for reductions
− No support for data management

```
78      DO CONCURRENT (k=y_min:y_max)
79        DO CONCURRENT (j=x_min:x_max) LOCAL(right_flux,left_flux,top_flux,bottom_flux,
                                              total_flux,min_cell_volume,energy_change,
                                              recip_volume)
80
81          left_flux=  (xarea(j  ,k  )*(xvel0(j  ,k  )+xvel0(j  ,k+1)                  &
82                                      +xvel0(j  ,k  )+xvel0(j  ,k+1)))*0.25_8*dt*0.5
83          right_flux= (xarea(j+1,k  )*(xvel0(j+1,k  )+xvel0(j+1,k+1)                  &
84                                      +xvel0(j+1,k  )+xvel0(j+1,k+1)))*0.25_8*dt*0.5
85          bottom_flux=(yarea(j  ,k  )*(yvel0(j  ,k  )+yvel0(j+1,k  )                  &
86                                      +yvel0(j  ,k  )+yvel0(j+1,k  )))*0.25_8*dt*0.5
87          top_flux=   (yarea(j  ,k+1)*(yvel0(j  ,k+1)+yvel0(j+1,k+1)                  &
88                                      +yvel0(j  ,k+1)+yvel0(j+1,k+1)))*0.25_8*dt*0.5
89          total_flux=right_flux-left_flux+top_flux-bottom_flux
90
91          volume_change(j,k)=volume(j,k)/(volume(j,k)+total_flux)
92
93          min_cell_volume=MIN(volume(j,k)+right_flux-left_flux+top_flux-bottom_flux &
94                             ,volume(j,k)+right_flux-left_flux                      &
95                             ,volume(j,k)+top_flux-bottom_flux)
97          recip_volume=1.0/volume(j,k)
99          energy_change=(pressure(j,k)/density0(j,k)+viscosity(j,k)/density0(j,k))*...
101         energy1(j,k)=energy0(j,k)-energy_change
103         density1(j,k)=density0(j,k)*volume_change(j,k)
105       ENDDO
106     ENDDO
107
```

PGI