Scientific and Technical Computing

Hardware and Code Optimization

Lars Koesterke

UT Austin, 10/22/19

Compiling Code

Generation of ‘Optimization Reports’

Compiling Code

```
! compile: ifort -c -qopt-report=3 -qopt-report-phase=vec add.f90

subroutine add(a, b, c, n)
real, dimension(n) :: a, b, c
do i=1, n                              ! Line 5
  a(i) = a(i) + b(i) + c(i)
enddo
do i=1, n, 2                           ! Line 8
  a(i) = a(i) + b(i) + c(i)
enddo
do i=1, n                              !Line 11
  a(i) = a(i-1) + b(i) + c(i)
enddo
end subroutine add
```

Compiling Code

```
LOOP BEGIN at add.f90(5,1)
   remark #15300: LOOP WAS VECTORIZED
   remark #15442: entire loop may be executed in remainder
   remark #15448: unmasked aligned unit stride loads: 2
   remark #15449: unmasked aligned unit stride stores: 1
   remark #15450: unmasked unaligned unit stride loads: 1
   remark #15475: --- begin vector cost summary ---
   remark #15476: scalar cost: 9
   remark #15477: vector cost: 2.000
   remark #15478: estimated potential speedup: 4.180
   remark #15488: --- end vector cost summary ---
LOOP END
```

Compiling Code

```
LOOP BEGIN at add.f90(8,1)
   remark #15335: loop was not vectorized: vectorization possible but
seems inefficient. Use vector always directive or -vec-threshold0 to
override
   remark #15452: unmasked strided loads: 3
   remark #15453: unmasked strided stores: 1
   remark #15475: --- begin vector cost summary ---
   remark #15476: scalar cost: 9
   remark #15477: vector cost: 36.750
   remark #15478: estimated potential speedup: 0.240
   remark #15488: --- end vector cost summary ---
LOOP END
```

Compiling Code

```
LOOP BEGIN at add.f90(11,1)
   remark #15344: loop was not vectorized: vector dependence prevents
vectorization. First dependence is shown below. Use level 5 report for details
   remark #15346: vector dependence: assumed FLOW dependence between a(i) (12:3)
and a(i-1) (12:3)
LOOP END
```