Lecture 15 - Performance Tuning Examples

Prof. Brendan Kochunas

October 21, 2020

Contents

These examples are intended to be inspected using the Compiler Explorer. Alternatively they could be done with any Fortran compiler and the `-S` and `-o` options to output the generated assembly.

1 Squaring

Use no compiler options. Then use `-O3`, `-Ofast`, `-Ofast -mavx512f`

```
real(8) function square(x)
    implicit none
    real(8), intent(in) :: x
    square = x * x * x  * x
    return
end function square


real(8) function square2(x)
    implicit none
    real(8), intent(in) :: x
    square2 = x**4
    return
end function square2

real(8) function square3(x)
    implicit none
    real(8), intent(in) :: x
    square3 = x**(4.0)
    return
end function square3
```

2 Exponentiation

Use no compiler options. Then use `-O3`, `-Ofast`, `-Ofast -mavx512f`

```
real(8) function exp1(a,b)
    implicit none
    real(8), intent(in) :: a,b
    exp1 = a**b
    return
end function exp1

real(8) function exp2(a,b)
    implicit none
    real(8), intent(in) :: a,b
    exp2 = EXP(b*LOG(a))
    return
end function exp2
```

3 Repeated Division

```
subroutine div1(a,b,c,n)
    implicit none
    integer, intent(in) :: n
    real(8), intent(in) :: b(*),c
    real(8), intent(out) :: a(*)

    integer :: i

    do i=1,n
      a(i)=b(i)/c
    end do

    return
end subroutine div1


subroutine div2(a,b,c,n)
    implicit none
    integer, intent(in) :: n
    real(8), intent(in) :: b(*),c
    real(8), intent(out) :: a(*)

    integer :: i
    real(8) :: rc

    rc=1.0d0/c
    do i=1,n
      a(i)=b(i)*c
    end do

    return
end subroutine div2
```

4 Polynomial Evaluation

Use no compiler options. Then use `-O3`, `-Ofast`, `-Ofast -mavx512f`

```
real(8) function poly1(a0,a1,a2,a3,a4,x)
    real(8),intent(in) :: a0,a1,a2,a3,a4,x
    poly1 = a0 + a1*x + a2*x**2 + a3*x**3 + a4*x**4
    return
end function poly1

real(8) function poly2(a0,a1,a2,a3,a4,x)
    real(8),intent(in) :: a0,a1,a2,a3,a4,x
    poly2 = a0 + x*(a1 + x*(a2 + x*(a3 + x*a4)))
    return
end function poly2
```

5 Loop Unrolling

Use no compiler options. Then use `-O3`, `-Ofast`, `-Ofast -mavx512f` and `-funroll-all-loops`

```
subroutine lngth1(n,a,s)
  integer :: n
  real(8) :: s,a(n)
  integer :: i
  s=0.d0
  do i=1,n
    s=s+a(i)*a(i)
  enddo
endsubroutine

subroutine lngth4(n,a,s)
  integer :: n
  real(8) :: s,a(n)
  integer :: i
  real(8) :: t1,t2,t3,t4

  t1=0.d0; t2=0.d0; t3=0.d0; t4=0.d0
  do i=1,n-3,4
    t1=t1+a(i)*a(i)
    t2=t2+a(i+1)*a(i+1)
    t3=t3+a(i+2)*a(i+2)
    t4=t4+a(i+3)*a(i+3)
  enddo
  s=t1+t2+t3+t4
endsubroutine
```