Introduction to High-Performance Computing

Exercise/4

Giorgio Amati

Corso di dottorato in Ingegneria Aeronautica e Spaziale 2024

g.amati@cineca.it / g.amaticode@gmail.com

Agenda

- ✓ Stress the compiler performance using MM
- ✓ Use standard “naive MM” (no blocking/no unrolling)
- ✓ size > 2048
  - use (if available) different compilers
  - use (if available) different HW
  - different options

Homework: fill the table

Results, in Mflops,

	gfortran	nvfortran	ifx
-O0	-	-	-
-O1	-	-	-
-O2	-	-	-
-O3	-	-	-
-fast	-	-	-
???	-	-	-

Hint

✓ Use, if possible, compiler flags like -Minfo (Nvidia)

Homework: how to improve the result?

- ✓ Computing pi

```
program pigreco
     implicit none
     integer :: i, elements
     real :: sum1, sum2
     real :: pi
     real(kind(1.d0)), parameter :: PI25DT = acos(-1.d0)
     write(6,*) "Elements?"
     read(5,*) elements
!
     sum1=0.0
     sum2=0.0
!
```

Homework: how to improve the result?

```
! forward sum...
    do i=1,elements
    sum1=sum1+(1.0/(float(i)*float(i)))
    end do
!
! backward sum…
    do i=elements, 1, -1
    sum2=sum2+(1.0/(float(i)*float(i)))
    end do
!
    call cpu_time(time2)

    write(6,*) 'The True PI =', PI25DT
    write(6,*) 'Computed PI (forward) =', sqrt(6.0*sum1),sqrt(6.0*sum1)/PI25DT
    write(6,*) 'Computed PI (backward)=', sqrt(6.0*sum2),sqrt(6.0*sum2)/PI25DT
```