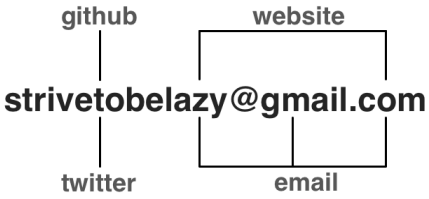
Presentation 05-Dec-2017

QE-GPU, an experience of porting to GPUs



Anoop
strivetobelazy@gmail.com
MHPC (ICTP - SISSA)

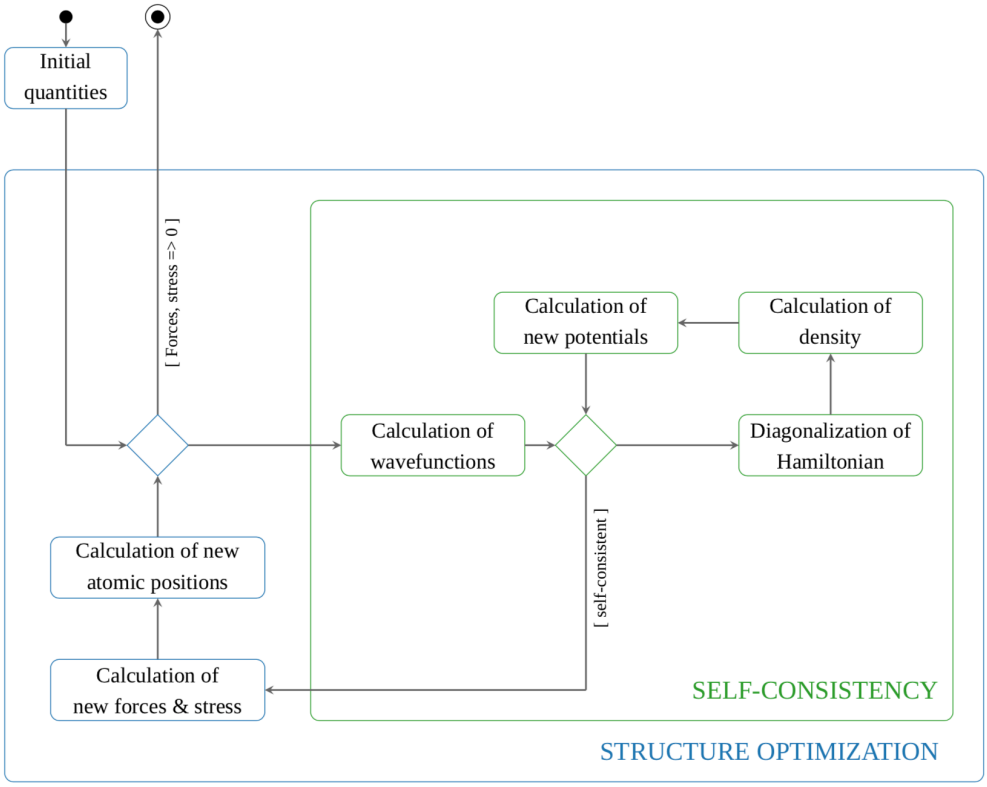
Dr Stefano de Gironcoli
degironc@sissa.it
SISSA

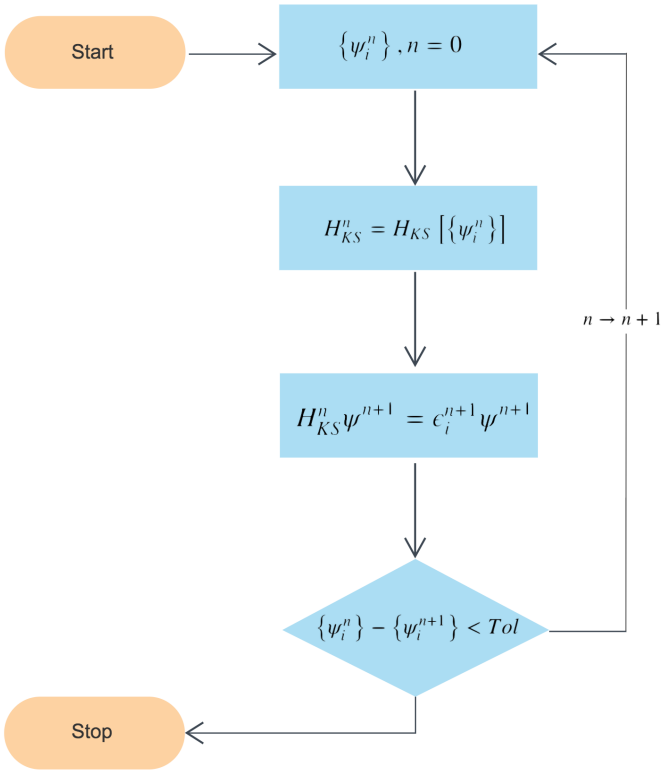
Ivan Girotto
igirotto@ictp.it
ICTP

Filippo Spiga
spiga.filippo@gmail.com
Cambridge

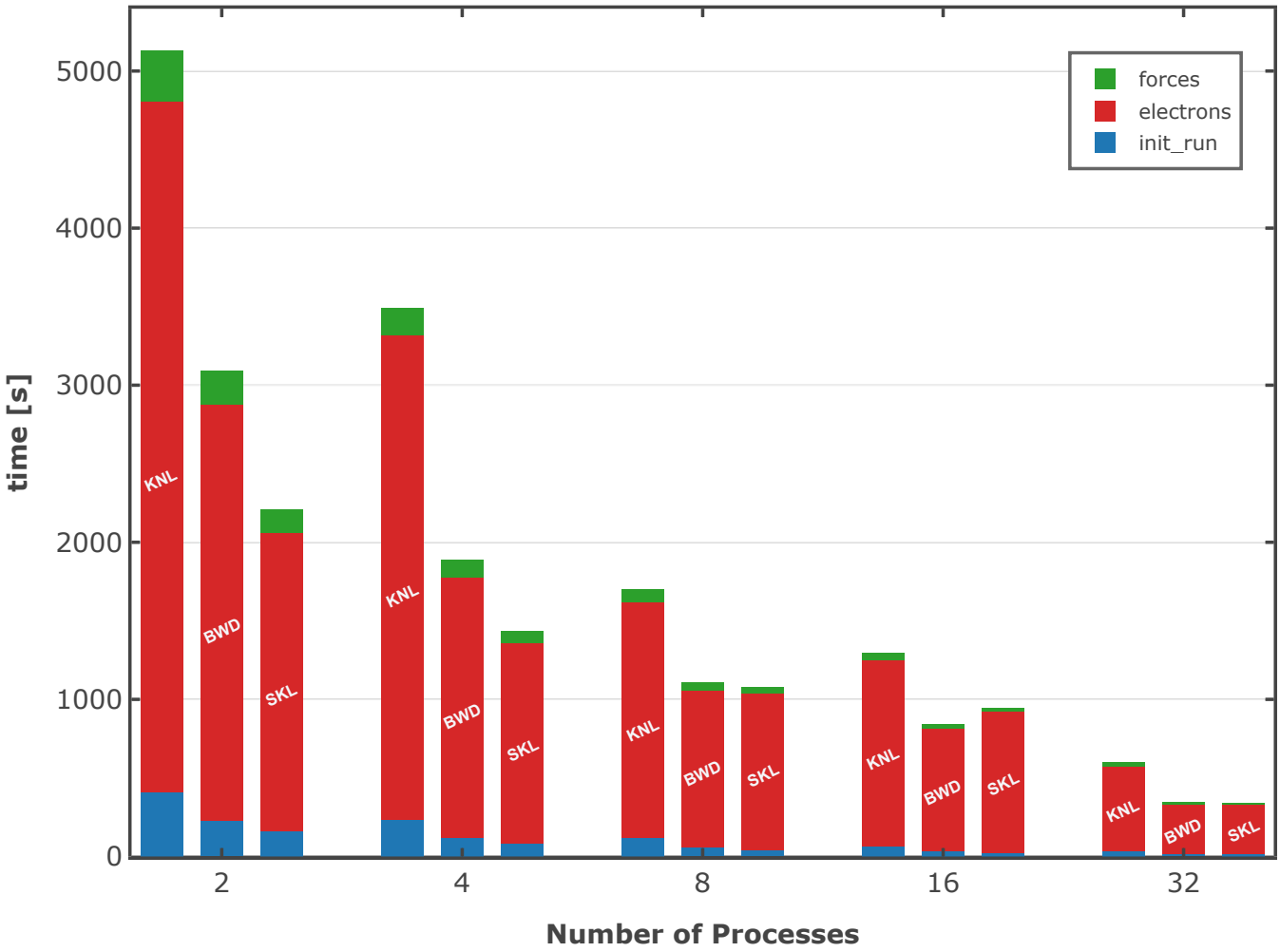
If things go by plan

- Algorithm you have seen a million times
- Background of the problem
- Proposed solutions to this problem
- How close are we to the solution?
- What you can gain from this boring presentation (CUDA Fortran)
- A sigular code CPU & GPU solution to exascale QE

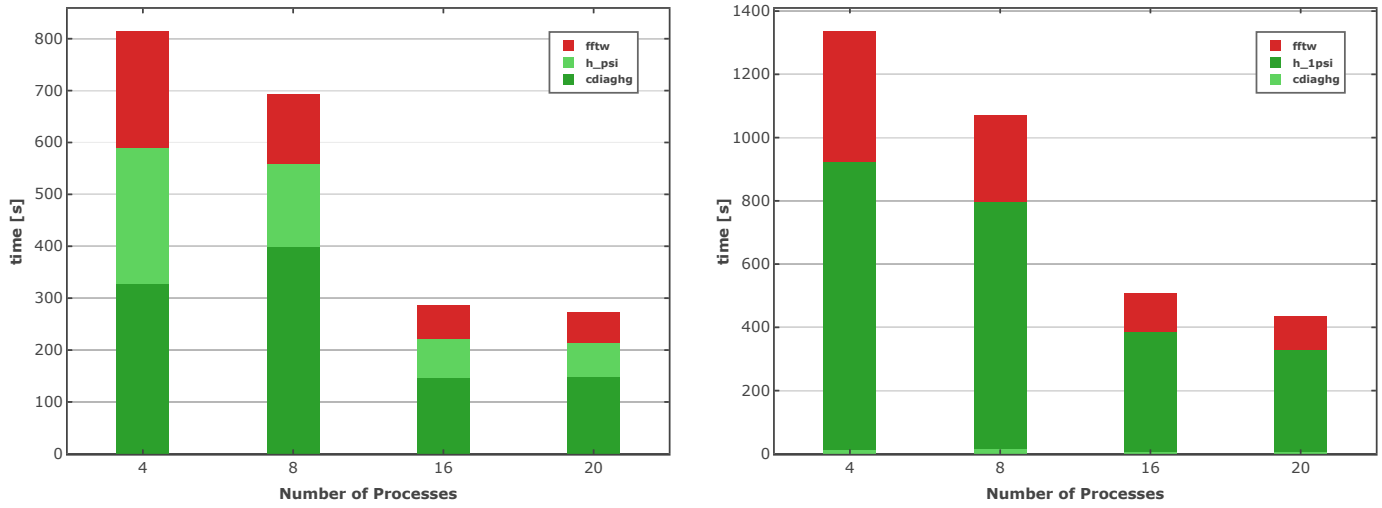




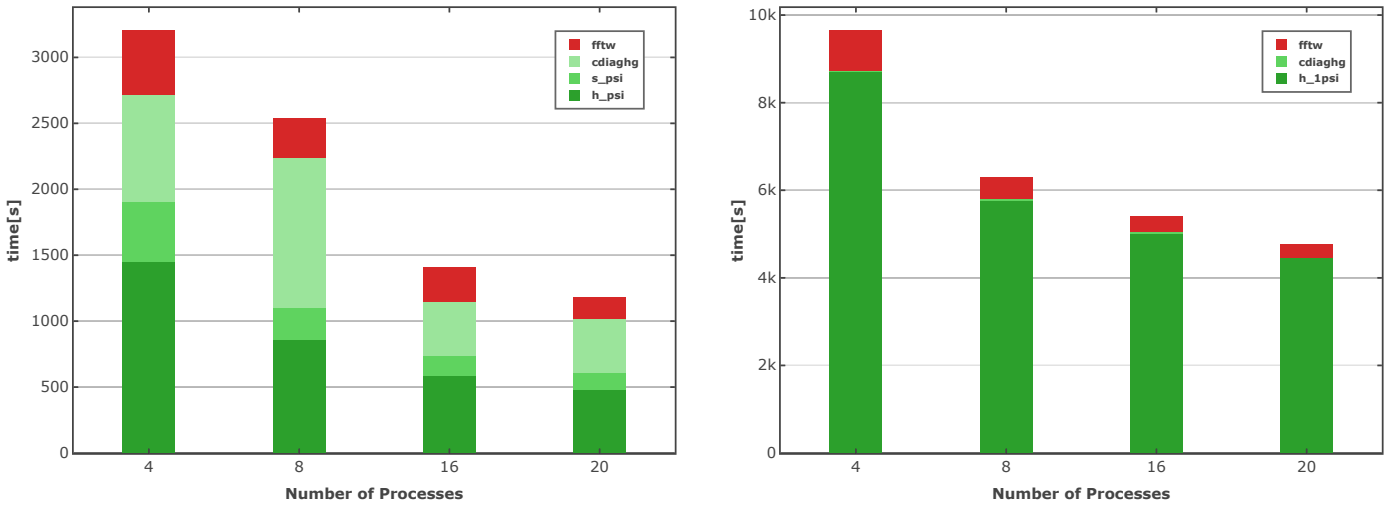
MPI Performance AUSURF112



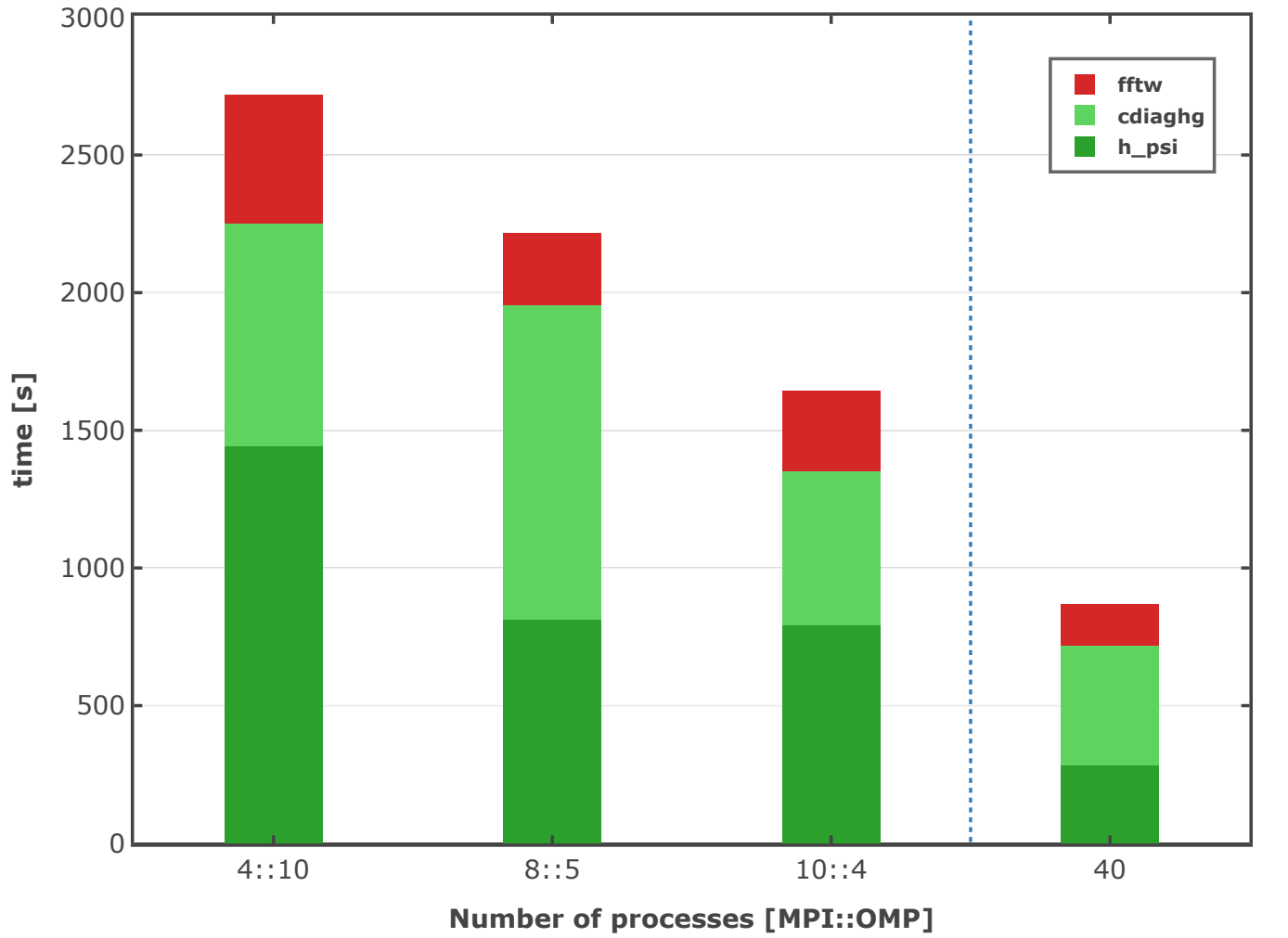
MPI Performance SiO2 Davidson (left) CG (right)

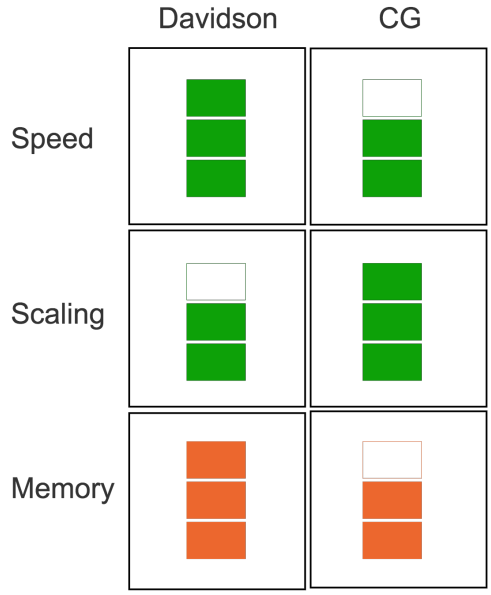


MPI Performance AUSURF112 Davidson (left) CG (right)



MPI+OMP Performance of Davidson (AUSURF)



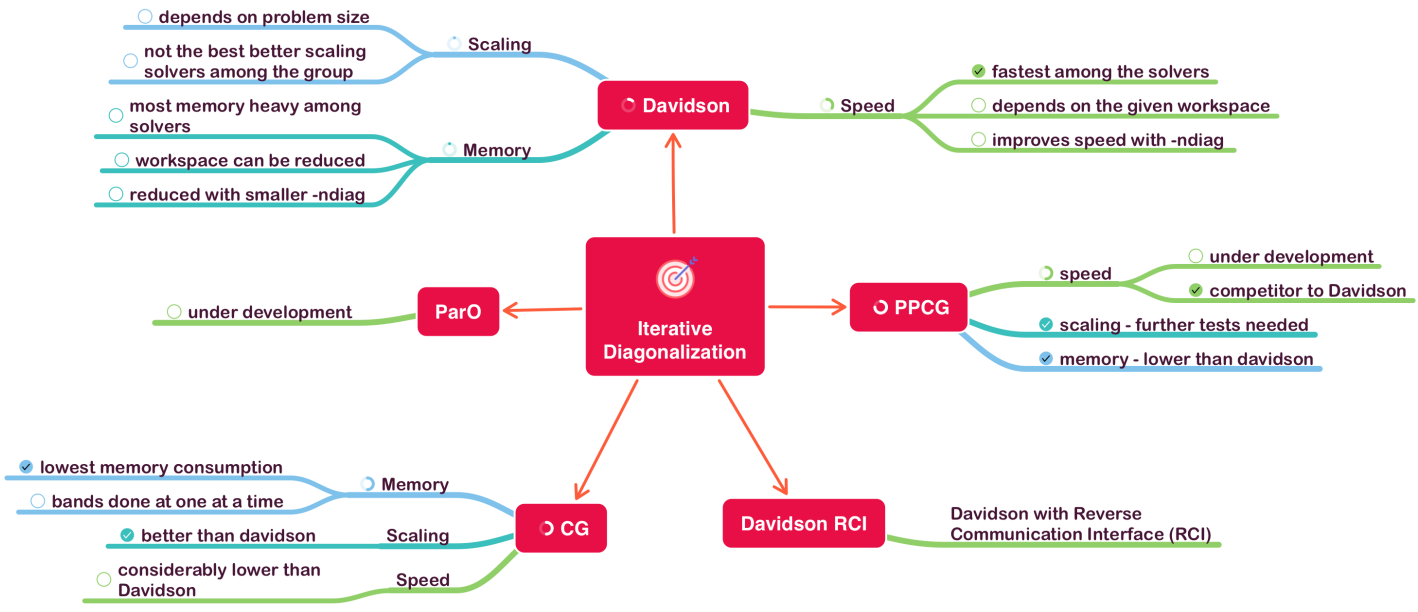


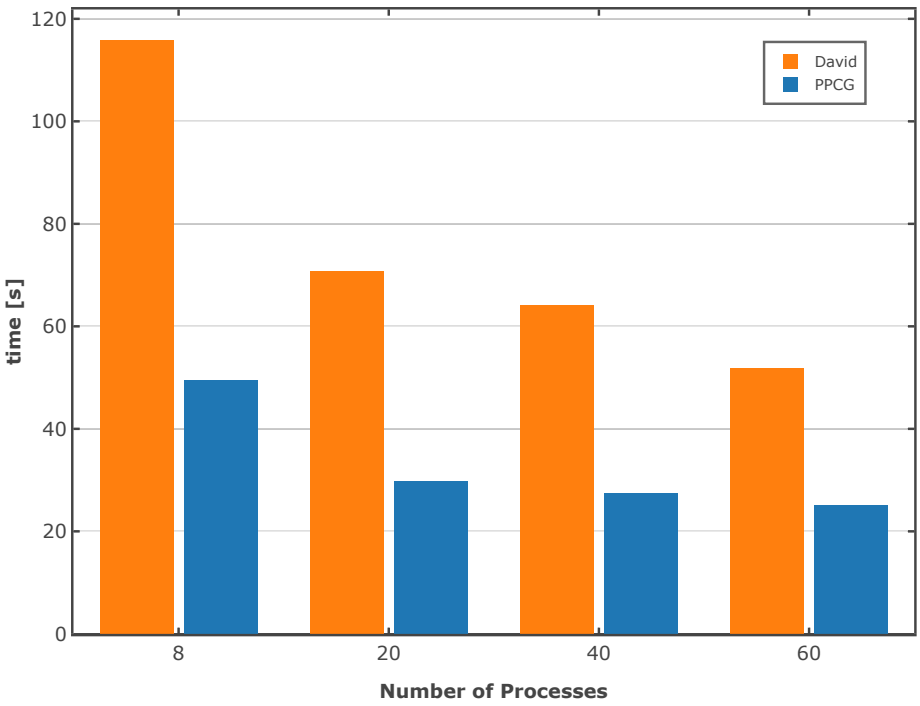
solutions

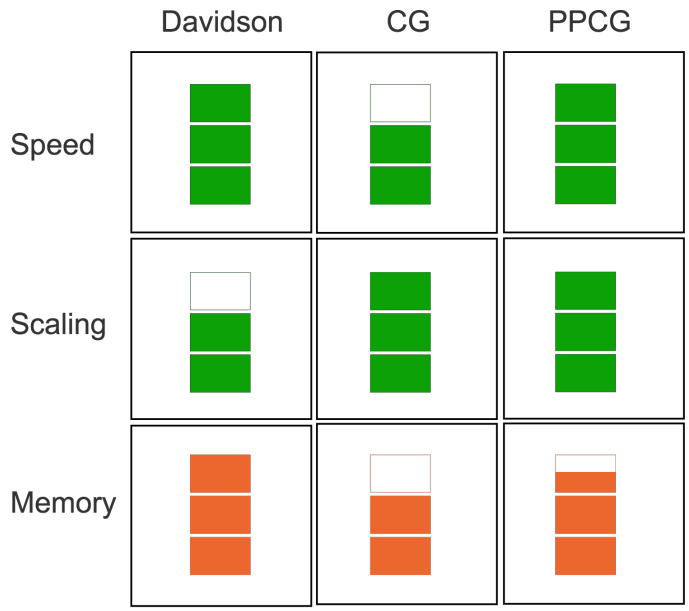
🧐 **Finding a better Method**

🤓 **Porting currunt code to GPU systems**

The ESLW project




Davidson vs PPCG Performance
120
100
80
60
40
20
0
time [s]
David
PPCG
8
20
40
60
Number of Processes


Davidson
CG
PPCG
Speed
Scaling
Memory

Solution 2: Porting to GPU systems

🙄 Declare and allocate host and device memory

😴 Initialize host data

😡 Transfer data from the host to the device

🤔 Execute one or more kernels

🤬 Transfer results from the device to the host

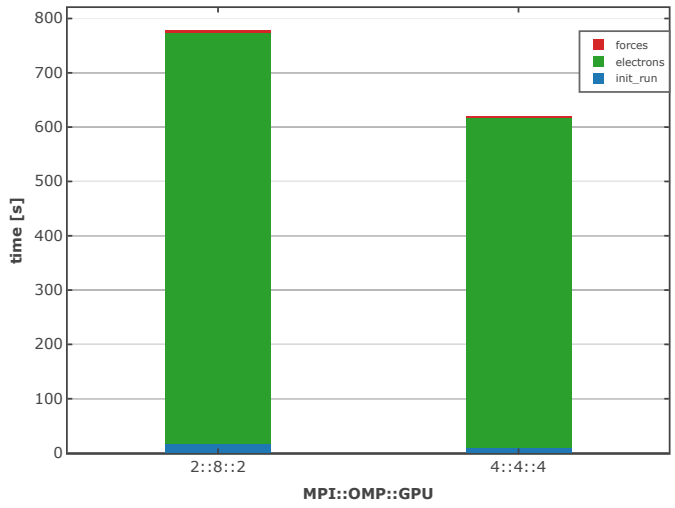
```
module lazy
contains
  subroutine whatever(N, x, y, a)
    implicit none
    integer,intent(in):: N
    real,intent(in) :: x(N), a
    real,intent(inout) :: y(N)
    integer :: i
    !
    do i = 1, N
      y(i) = y(i) + a*x(i)
    end do
    !
  end subroutine
end module

program mySGEVV
  use lazy, only:whatever
  implicit none
  integer,parameter:: N=40000000
  real :: x(N), a
  real :: y(N)
  integer :: i
  x = 1.0; y = 2.0; a = 2.0
  call whatever(N,x,y,a)
end program
```

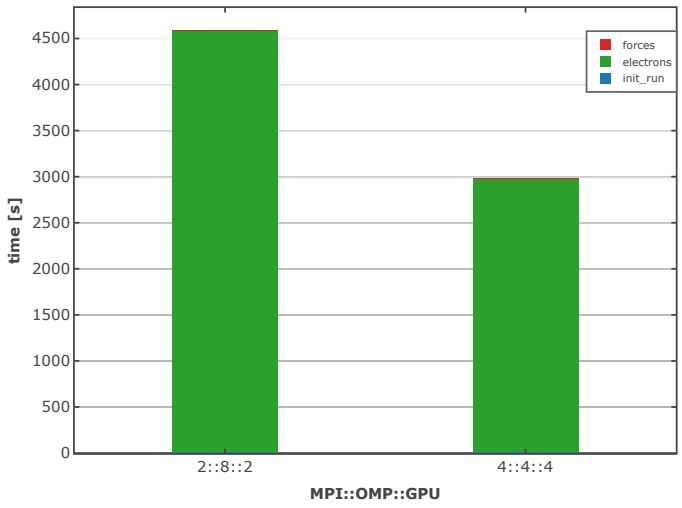
```
module lazy
contains
  attributes(global) subroutine whatever(x, y, a)
    implicit none
    real :: x(:), y(:); real, value :: a; integer :: i, n
    n = size(x); i = blockDim%x * (blockIdx%x - 1) + threadIdx%x
    if (i <= n) y(i) = y(i) + a*x(i)
  end subroutine
end module

program mySGEVV
  use lazy; use cudafor
  implicit none
  integer, parameter :: N = 40000000
  real :: x(N), y(N), a
  real, device :: x_d(N), y_d(N)                    ! Declare and allocate host
& device
  integer :: i

  type(dim3) :: grid, tBlock; tBlock = dim3(256,1,1)
  grid = dim3(ceiling(real(N)/tBlock%x),1,1)

  x = 1.0; y = 2.0; a = 2.0                         ! Initialize host data
  x_d = x; y_d = y                                  ! Transfer data from the ho
st to the device
  itr = 1000
  call whatever<<<grid, tBlock>>>(x_d, y_d, a)        ! Execute kernels
  y = y_d                                          ! Transfer results from the
device to the host
end program
```

```
module m
    ...
    real :: a(n)
    real,device :: a_d(n)
    ...
end module

subroutine update
#ifdef USE_GPU
    use m, only: a => a_d
#else
    use m, only: a
#endif
    ...
!$cuf kernel do <<<*,*>>>
    do i=1, n
        a(i) = a(i) + b
    enddo
    ...
end subroutine update
```

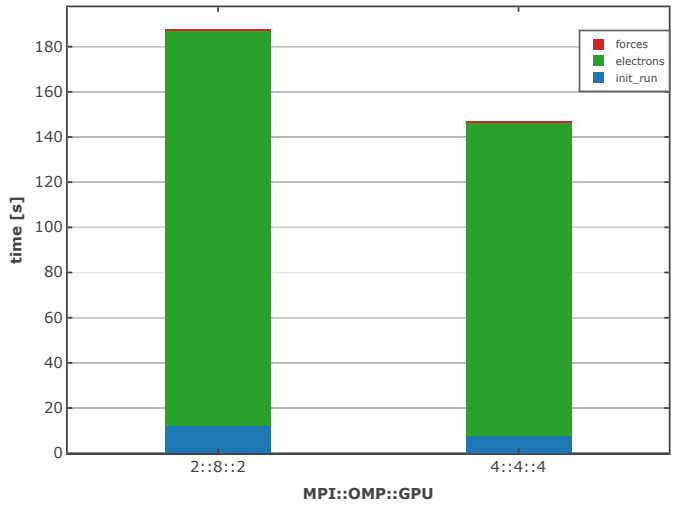
```
subroutine add_scalar(a,n)
    real:: a(n)
#ifdef USE_GPU
    attributes(device) :: a
#endif
!$cuf kernel do <<<*,*>>>
    do i=1, n
        a(i)=a(i)+b
    end do
end subroutine add_scalar
```

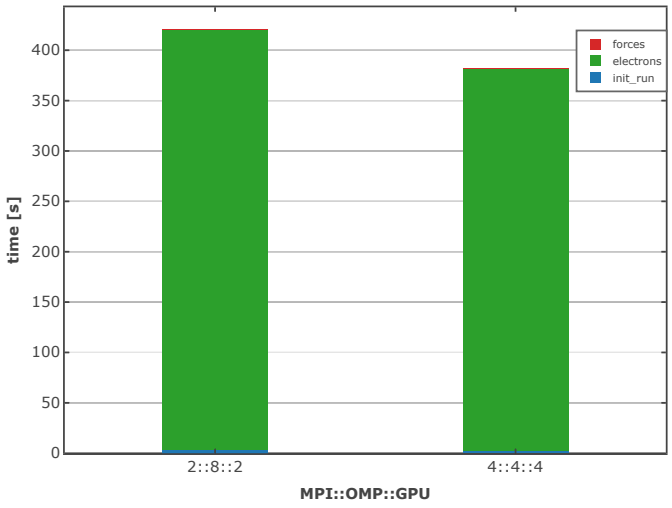
GPU Performance of AUSURF112 - Drake



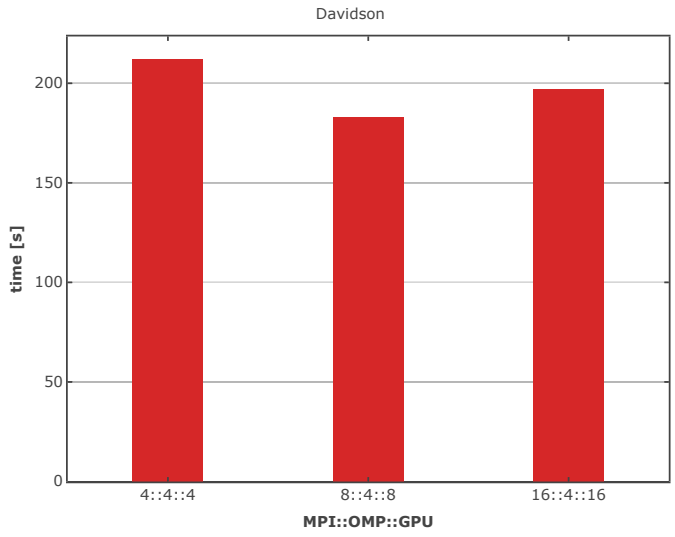


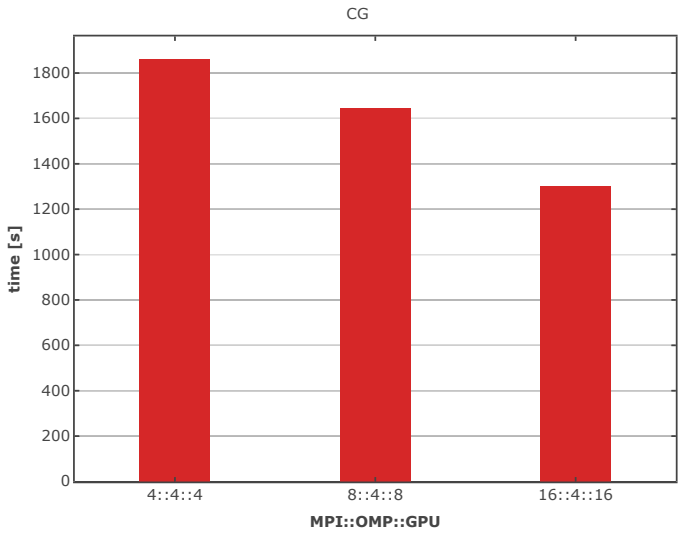
GPU Performance of SiO2 - Drake

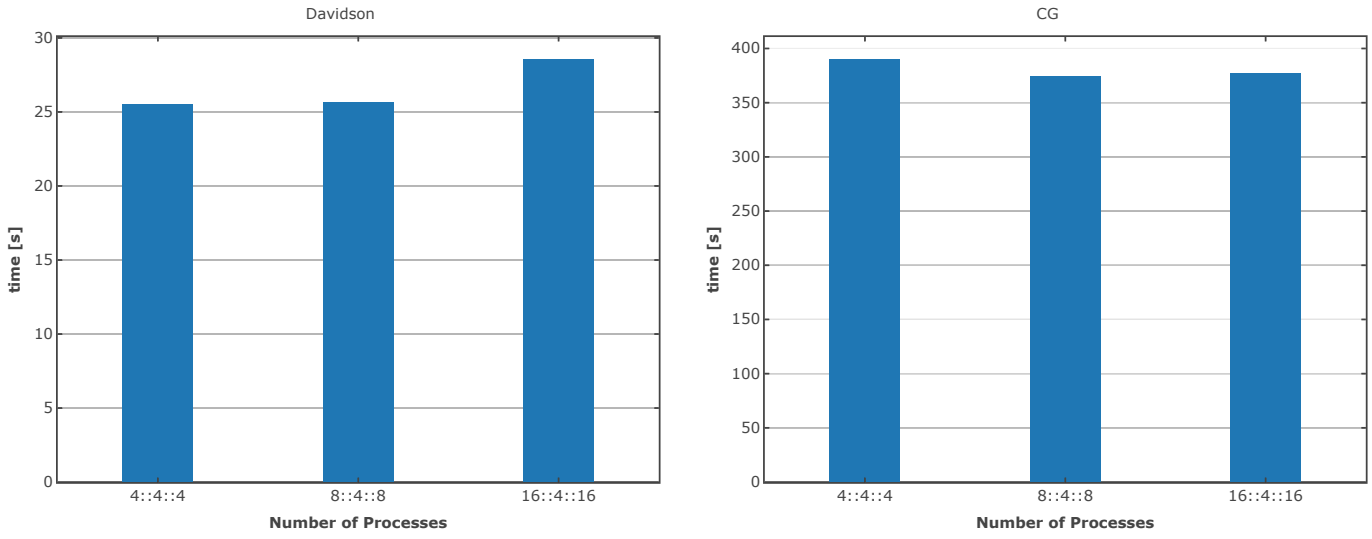


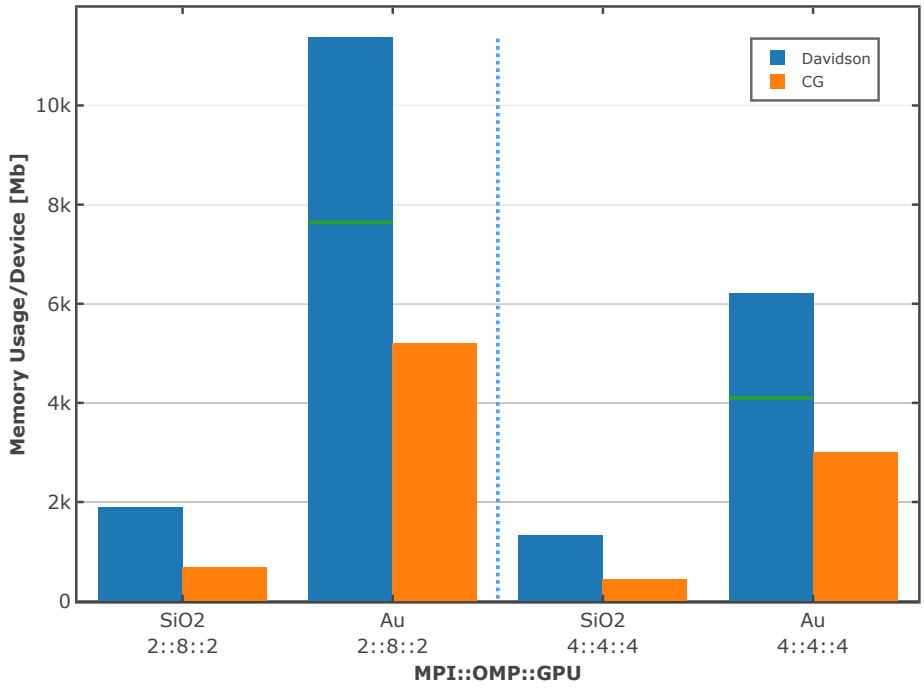


GPU Performance of AUSURF112 - DAVIDE

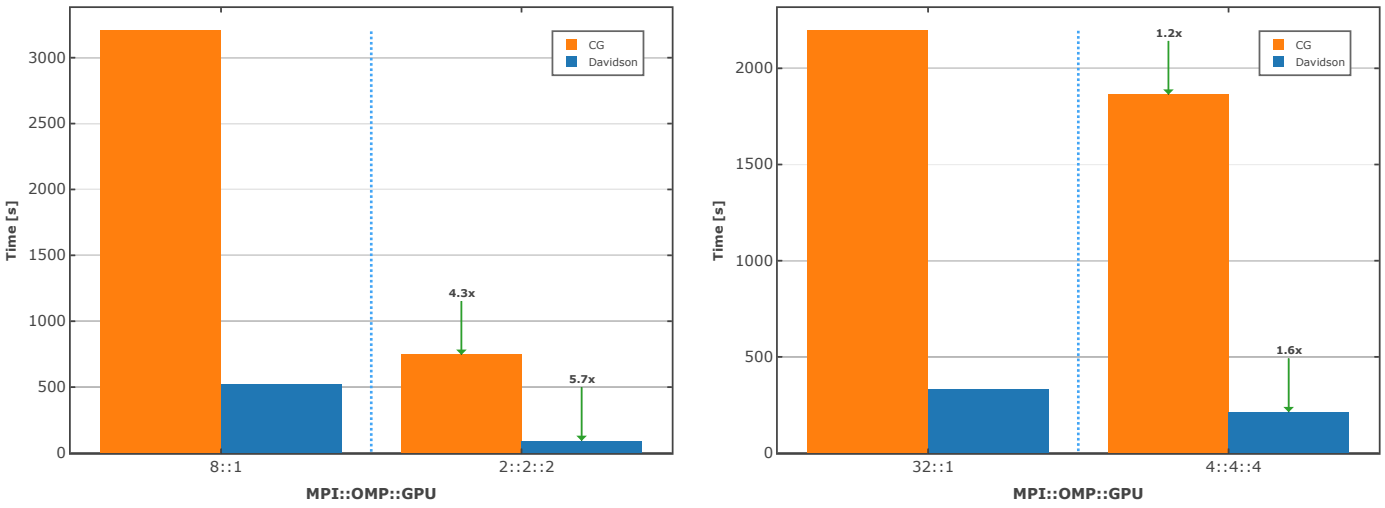




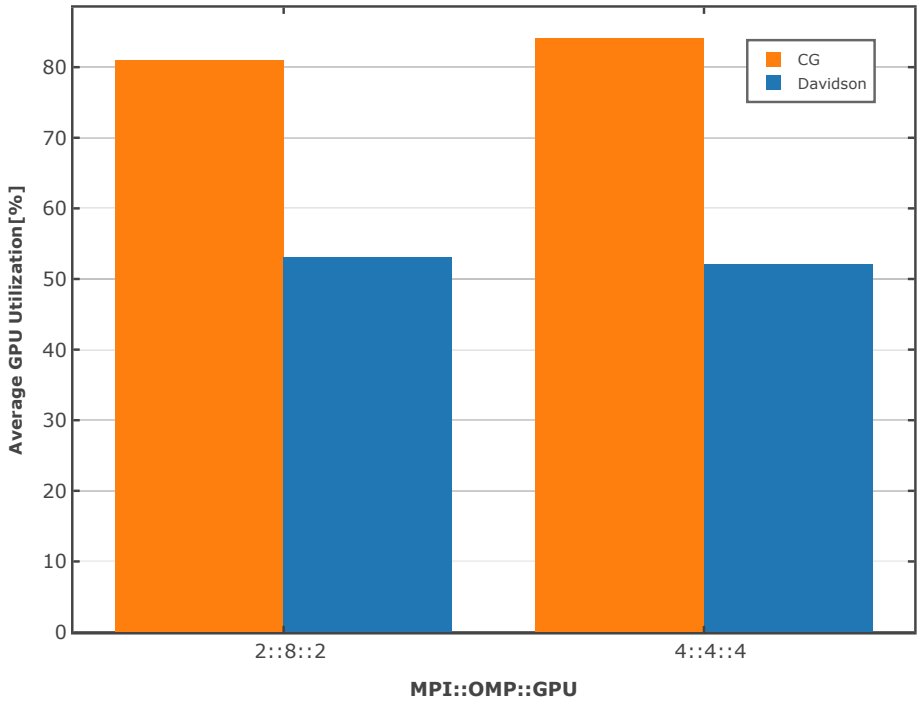

GPU Performance of SiO2 - DAVIDE
Davidson
CG
time [s]
Number of Processes
4::4::4
8::4::8
16::4::16


Memory Usage Davidson vs CG
Davidson
CG
Memory Usage/Device [Mb]
SiO2
2::8::2
Au
2::8::2
SiO2
4::4::4
Au
4::4::4
MPI::OMP::GPU

CPU vs GPU for Davidson and CG



Average Volatile GPU usage Davidson vs CG



Version 1.0 of QE-GPU is available at https://github.com/fspiga/qe-gpu (https://github.com/fspiga/qe-gpu)

contact me at strivetobelazy@gmai.com

Thank You