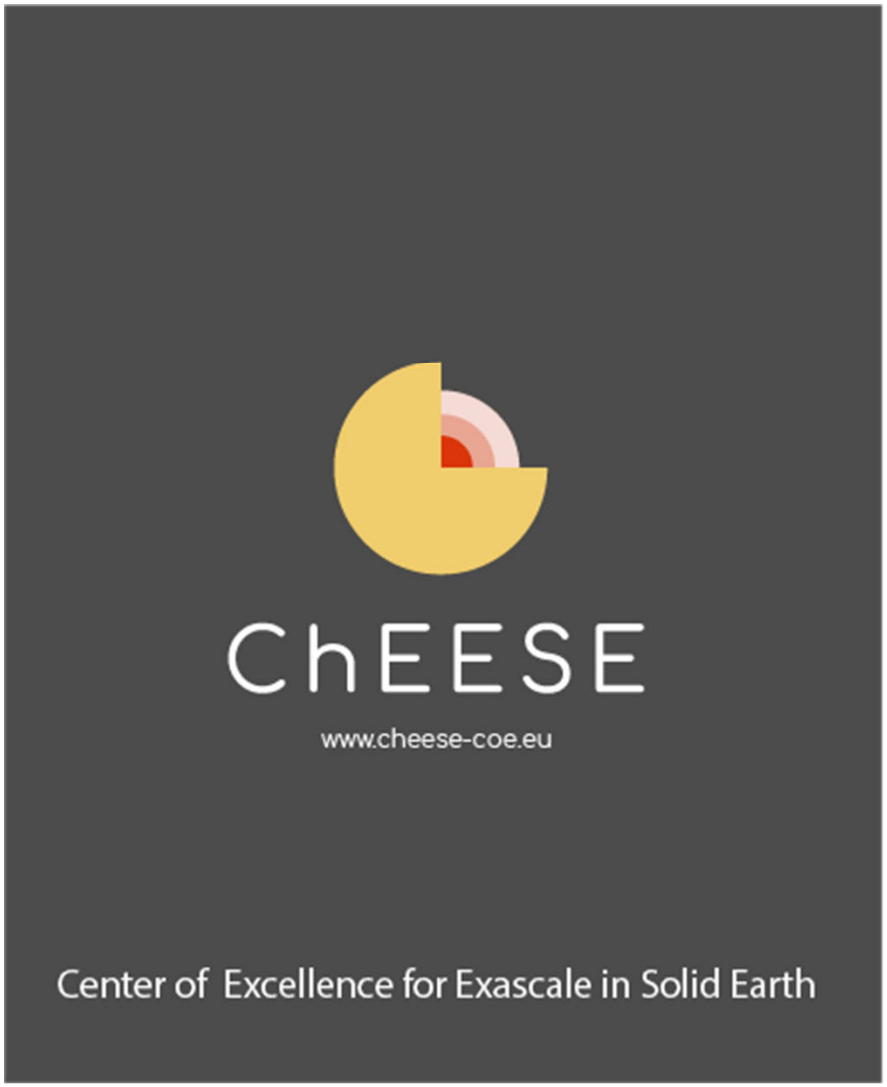


ACCelerating the FALL3D flagship code. Insights from porting a Mini-app

Eduardo Cabrera
Arnau Folch
Leonardo Mingari

BSC

Many thanks to our mentors:
Piero Lanucara (CINECA)
Lukas Mosimann (NVIDIA)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 823844

Outline

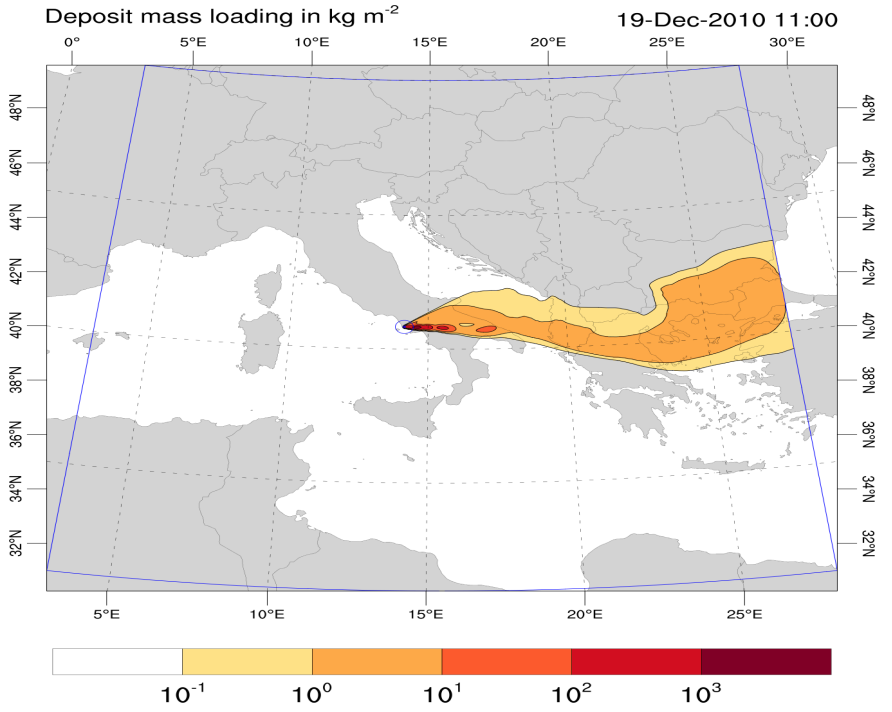
- FALL3D physics
- FALL3D components
- Implementation
  - Profiling
- FALL3D to fall3d
- ACCelerating
- Results
- Conclusions
- Future work

Outline

- **FALL3D physics**
- FALL3D components
- Implementation
  - Profiling
- FALL3D to fall3d
- ACCelerating
- Results
- Conclusions
- Future work

FALL3D

- ChEESE flagship code developed and maintained by BSC and INGV
- Eulerian model for the atmospheric transport and deposition of particles, aerosols and radionuclides
- Solves a set of advection-diffusion-sedimentation (ADS) equations on a structured grid using a second order finite volume explicit scheme (Euler or RK 4th in time)

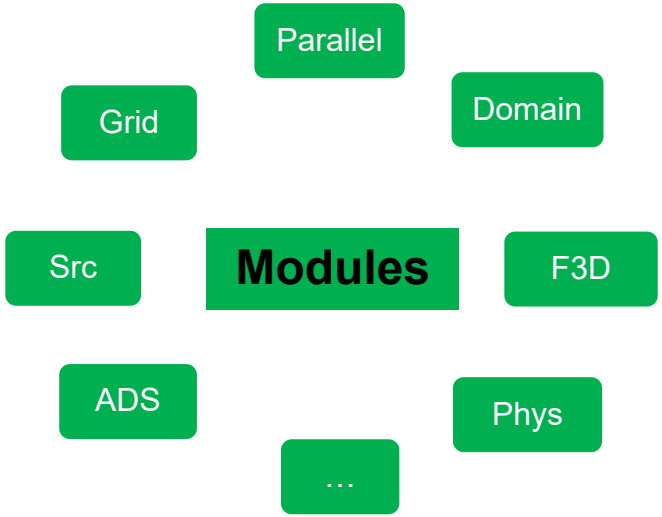


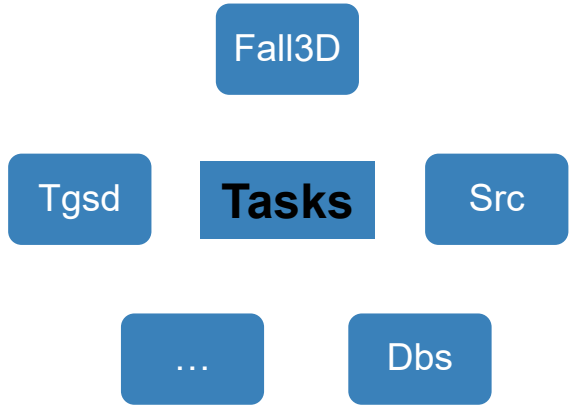
Outline

- FALL3D physics
- **FALL3D components**
- Implementation
  - Profiling
- FALL3D to fall3d
- ACCelerating
- Results
- Conclusions
- Future work

FALL3D

Components of FALL3D (~ 100K lines)





Outline

- FALL3D physics
- FALL3D components
- **Implementation**
  - Profiling
- FALL3D to fall3d
- ACCelerating
- Results
- Conclusions
- Future work

Implementation

- HPC main features (CPU implementation)
  - Implemented using Fortran 90
  - Only pure MPI parallelization for domain decomposition
  - Porting to accelerators (this talk)
  - OpenMP introduced at some regions of the code

Porting and Optimizing

- Why are we using OpenACC (More Science, Less Programming)?
  - ✓ Not too intrusive
  - ✓ Based on directives
  - ✓ *Easy* of maintenance => only one source code

- What is our favorite feature?
  - ✓ Flexibility, Portability & Performance

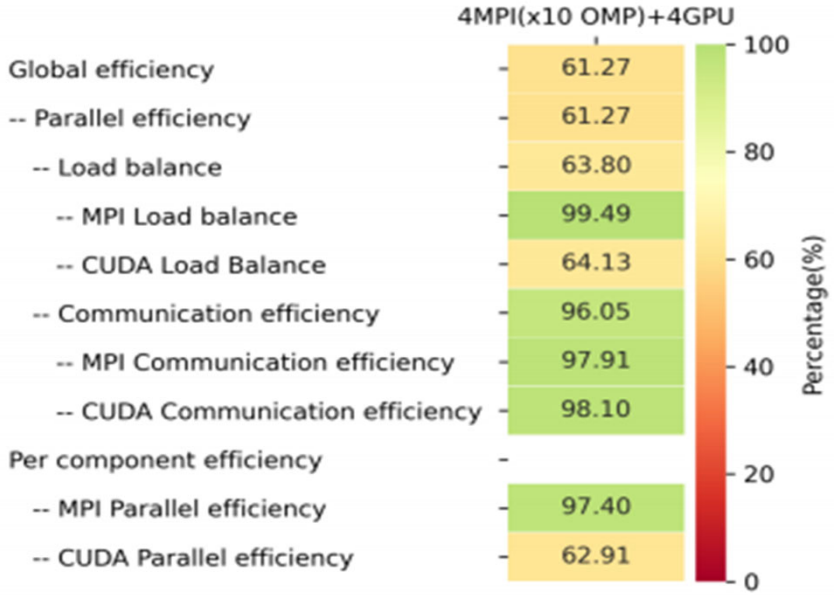
Porting and Optimizing

- MPI + OpenACC + OpenMP
  - Porting based on OpenACC GPU HACKATHON CINECA NVIDIA (2019)
  - Initially, porting has been limited to task_Fall3D & only in the mod_ADS (Runge-Kutta 4th order)
    - Trade off between computing and memory requirements
    - Implied a full restructuring of arrays and their access
- Tested @ BSC CTE Power
  - 2 x IBM Power9 8335-GTH + 4 x GPU NVIDIA V100 per node

FALL3D Performance Results

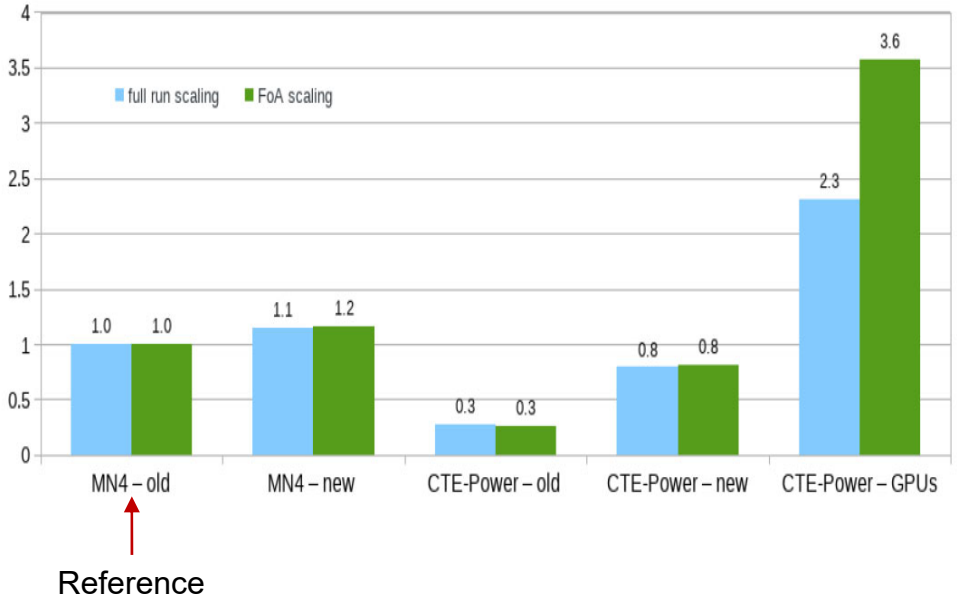
- Ported version audited by POP in the ChEESE WP2 framework
- At MPI level the code is very efficient and its load balance is almost perfect
- The efficiency analysis clearly points to the CUDA parallelization and more concretely to its load balance
- Currently GPUs are computing only ~ 27% of the iterative loop time

hybrid POP metrics (1 CTE node)

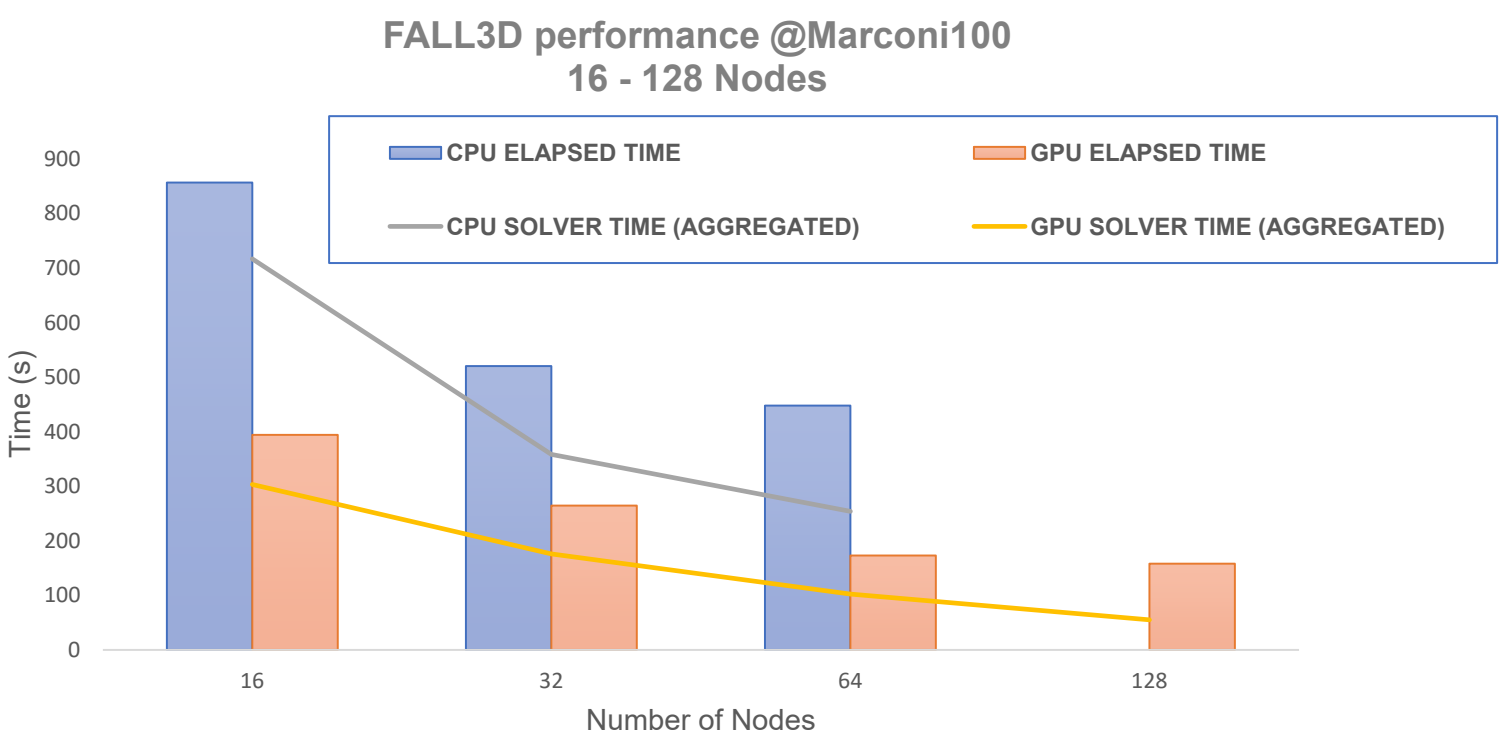


FALL3D Performance Results

- Currently GPUs are computing only ~ 27% of the iterative loop time.
- The OpenMP parallelization does not reduce enough the CPU time
- However, the GPU configuration is 3.6x faster than MN4 even in its current state of development



FALL3D Performance Results



Outline

- FALL3D physics
- FALL3D components
- Implementation
  - **Profiling**
- FALL3D to fall3d
- ACCelerating
- Results
- Conclusions
- Future work

FALL3D Profiling

Score-P Scalable performance measurement infrastructure for parallel codes **@Marconi100**

type	max_buf[B]	visits	time[s]	time[%]	time/visit[us]	region
ALL	1489393320	134861202	58336.11	100	432.56	ALL
OMP	104172160	35973120	50787.82	**87.1**	**1411.83**	**OMP**
MPI	63582881	13887410	3431.29	**5.9**	**247.08**	**MPI**
OPENACC	26317954	16195664	2626.01	**4.5**	**162.14**	**OPENACC**
USR	1294426926	53192656	767.37	1.3	14.43	USR
COM	29936010	15612336	723.61	1.2	46.35	COM
SCOREP	41	16	0	0	137.12	SCOREP

FALL3D Profiling

Score-P Scalable performance measurement infrastructure for parallel codes **@Marconi100**

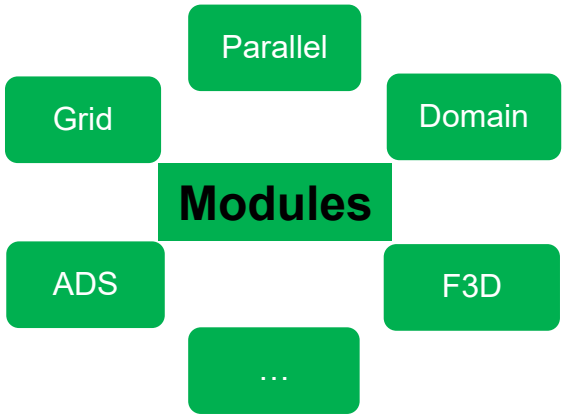
type	max_buf[B]	visits	time[s]	time[%]	time/visit[us]	region
OMP	17981184	11065344	31772.82	**54.5**	**2871.38**	**!$omp do @mod_ADS.F90:1050**
OMP	1498432	922112	13337.64	**22.9**	**14464.23**	**!$omp do @mod_F3D.f90:1406**
MPI	9132864	4215312	2884.44	**4.9**	**684.28**	**MPI_Wait**
OMP	832	512	2418.06	**4.1**	**4722777**	**!$omp do @mod_Phys.f90:619**
OMP	17981184	11065344	1914.15	**3.3**	**172.99**	**!$omp implicit barrier @mod_ADS.F90:1057**
OMP	1498432	922112	851.12	**1.5**	**923.02**	**!$omp implicit barrier @mod_F3D.f90:1425**
USR	3371472	2074752	744.46	1.3	358.82	ads_freeflow_
OMP	832	512	478.04	**0.8**	**933669.05**	**!$omp implicit barrier @mod_Phys.f90:637**
OPENACC	280956	172896	197.05	**0.3**	**1139.7**	**acc_wait@mod_ADS.F90:508**
COM	1126190	693040	170.21	**0.3**	**245.6**	**domain_domain_swap_mass_points_2halo_x_**

Outline

- FALL3D physics
- FALL3D components
- Implementation
  - Profiling
- **FALL3D to fall3d**
- ACCelerating
- Results
- Conclusions
- Future work

FALL3D to fall3d GPU HACKATHON CINECA NVIDIA (2021)

- The Mini-app FALL3D only solves task_Fall3D assuming a fixed problem (helicoidal velocity field)
- Components of fall3d (~ 10 K lines) & ~ 90% computing time of FALL3D
- MPI + OpenACC More Science, Less Programming + OpenMP



Outline

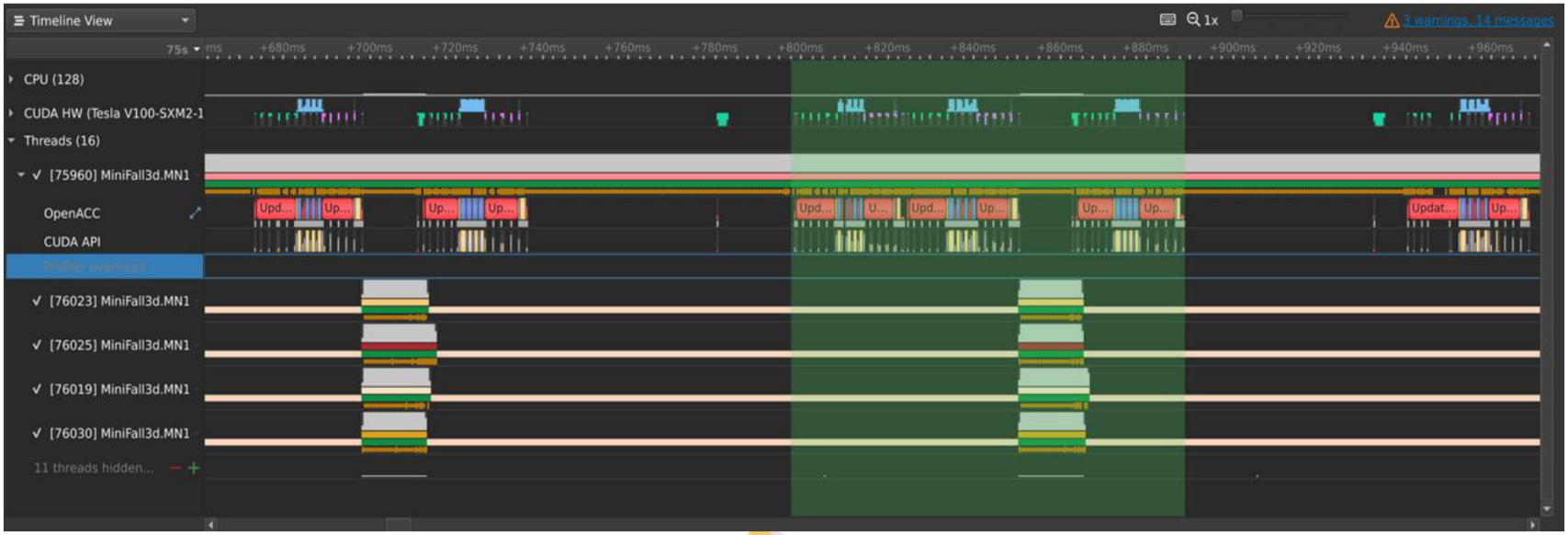
- FALL3D physics
- FALL3D components
- Implementation
  - Profiling
- FALL3D to fall3d
- **ACCelerating**
- Results
- Conclusions
- Future work

Accelerating fall3d using OpenACC

More Science, Less Programming

nsys profiling @M100 **hpc-sdk/2020--binary**

mpirun -np ${SLURM_NTASKS} --report-bindings -map-by socket:PE=8 nsys profile -f true -o out-r8-%q{OMPI_COMM_WORLD_RANK}-%q{OMPI_COMM_WORLD_SIZE} -t cuda,openacc ${BIN} ${FALLTASK} ${INPFILE} ${NX} ${NY} ${NZ}



Accelerating fall3d using OpenACC

More Science, Less Programming

- Increase GPUs' computing load (~ 27%)
  - Reorganize the code
  - Replacing all OpenMP with OpenACC

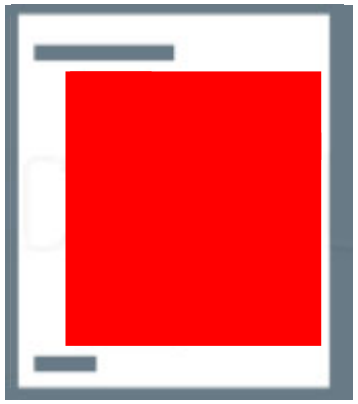
    3 do-loop cycles, specially one four nested loop

```
      do ibin = 1,MY_TRA%nbins
#if defined WITH_ACC
        call nvtxStartRange("F3D_time_step")
#endif
        !$acc parallel  default(none)
        !$acc loop gang
        do k = my_kps,my_kpe
           dZ = MY_GRID%dX3_p(k)
           !$acc loop worker
           do j = my_jps,my_jpe
              dY  = MY_GRID%dX2_p(j)
              Hm1 = MY_GRID%Hm1_p(j)
              Hm2 = MY_GRID%Hm2_p(j)
              !$acc loop vector
              do i = my_ips,my_ipe
                 dX  = MY_GRID%dX1_p(i)
                 Hm3 = MY_GRID%Hm3_p(i,j)
                 vol = dX*dY*dZ*(Hm1*Hm2*Hm3)
                    MY_TRA%my_c(i,j,k,ibin) = MY_TRA%my_c(i,j,k,ibin) + MY_TIME%gl_dt*MY_TRA%my_s(i,j,k,ibin)/vol
                 end do
              end do
           end do
        !$acc end parallel
#if defined WITH_ACC
        call nvtxEndRange()
#endif
      end do
```

Accelerating fall3d using OpenACC

More Science, Less Programming

- Fusion GPU regions inside 3 mod_ADS subroutines



- Fusion all device regions into **only one** large GPU region

Accelerating fall3d using OpenACC

More Science, Less Programming

- Fusion all device regions into **only one** large GPU region

```
!$acc data &
!$acc present(my_u,my_v) &
!$acc copyin(MY_MET,MY_MET%my_u,MY_MET%my_v,MY_MET%my_w) &
    !$acc copyin(MY_TRA) &
    !$acc copyin(MY_GRID) &
    !$acc copyin(MY_TIME) &
    !$acc copyin(MY_TRA%my_s,MY_TRA%nbins) &
    !$acc copyin(MY_TIME%gl_dt) &
    !$acc copyin(MY_GRID%dX3_p,MY_GRID%dX2_p,MY_GRID%dX1_p) &
    !$acc copyin(MY_GRID%Hm1_p,MY_GRID%Hm2_p,MY_GRID%Hm3_p) &
    !$acc copyout(MY_TRA%my_c) &
        !$acc copy(MY_TRA%my_c,MY_TRA%my_vs,MY_TRA%my_acum) &
        !$acc copy(MY_TRA%my_W_flux,MY_TRA%my_E_flux) &
        !$acc copy(MY_TRA%my_S_flux,MY_TRA%my_N_flux) &
        !$acc copy(MY_TRA%my_D_flux,MY_TRA%my_U_flux)
```

```
SUBROUTINE IRECV_REAL( array, n, isour, itag, ihand, what_group )
  IMPLICIT NONE
  REAL(rp)    :: array !array(*)
  INTEGER(ip) :: n, isour, itag, ihand, what_group
#if defined WITH_MPI
  INTEGER(ip) :: ierr
  !$acc host_data use_device(array) if_present
  CALL MPI_IRECV( array, n, MPI_PRECISION, isour, itag, what_group, ihand, ierr )
  !$acc end host_data
#endif
  RETURN
END SUBROUTINE IRECV_REAL
```

```
SUBROUTINE ISEND_REAL( array, n, idest, itag, ihand, what_group )
  IMPLICIT NONE
  REAL(rp)    :: array !array(*)
  INTEGER(ip) :: n, idest, itag, ihand, what_group
#if defined WITH_MPI
  INTEGER(ip) :: ierr
  !$acc host_data use_device(array) if_present
  CALL MPI_ISEND( array, n, MPI_PRECISION, idest, itag, what_group, ihand, ierr )
  !$acc end host_data
#endif
  RETURN
END SUBROUTINE ISEND_REAL
```

Porting and Optimizing

Issues & Requests

- GNU compiler version 8.4.0 (GCC) @M100 did not compile
- nsys profiling issue OpenMP does not show up @M100

  -t openmp

  NVIDIA Nsight Systems version 2021.1.1.66-6c5c5cb

- Deep copy in OpenACC Fortran (implicit attach behavior)
- CUDA-Aware MPI

Outline

- FALL3D physics
- FALL3D components
- Implementation
  - Profiling
- FALL3D to fall3d
- ACCelerating
- **Results**
- Conclusions
- Future work

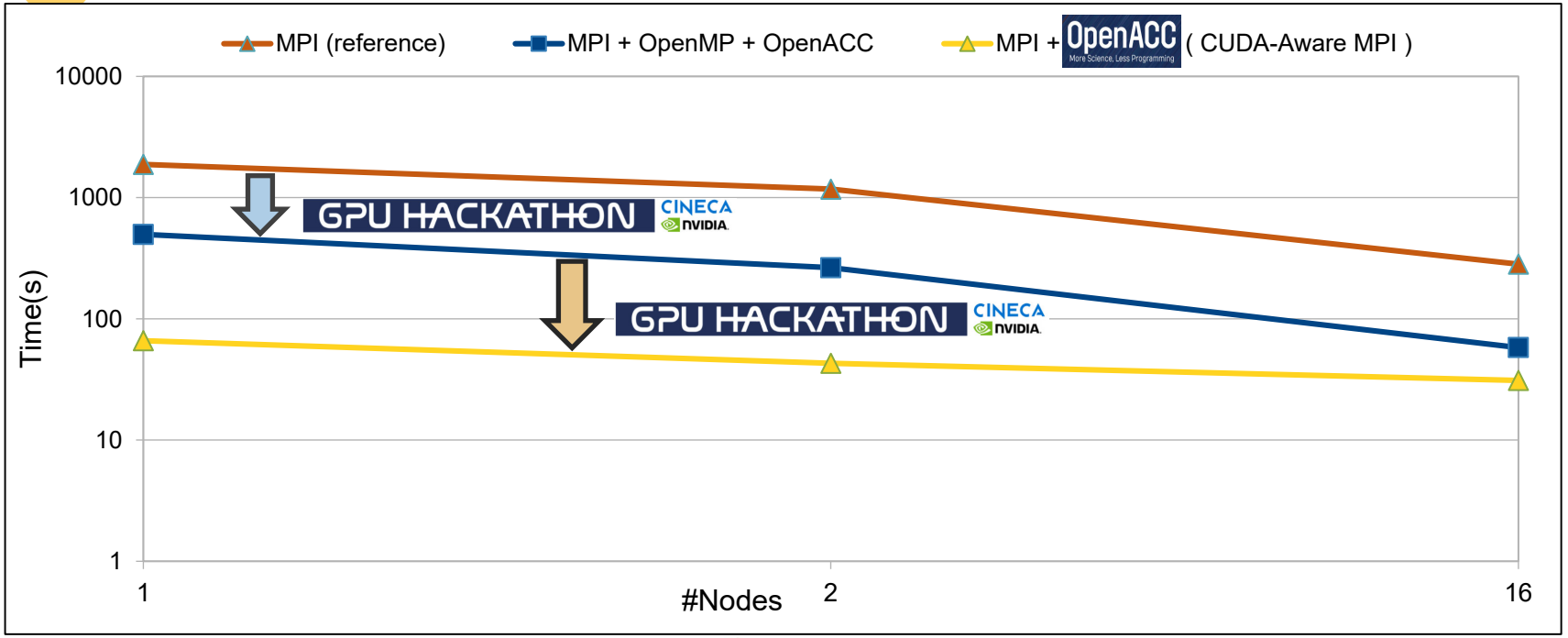
Test case

- GRID
  NX = 400
  NY = 400
  NZ = 240

- Domain decomposition nx, ny, nz
  1) 2, 2, 1
  2) 2, 2, 2
  3) 4, 4, 4

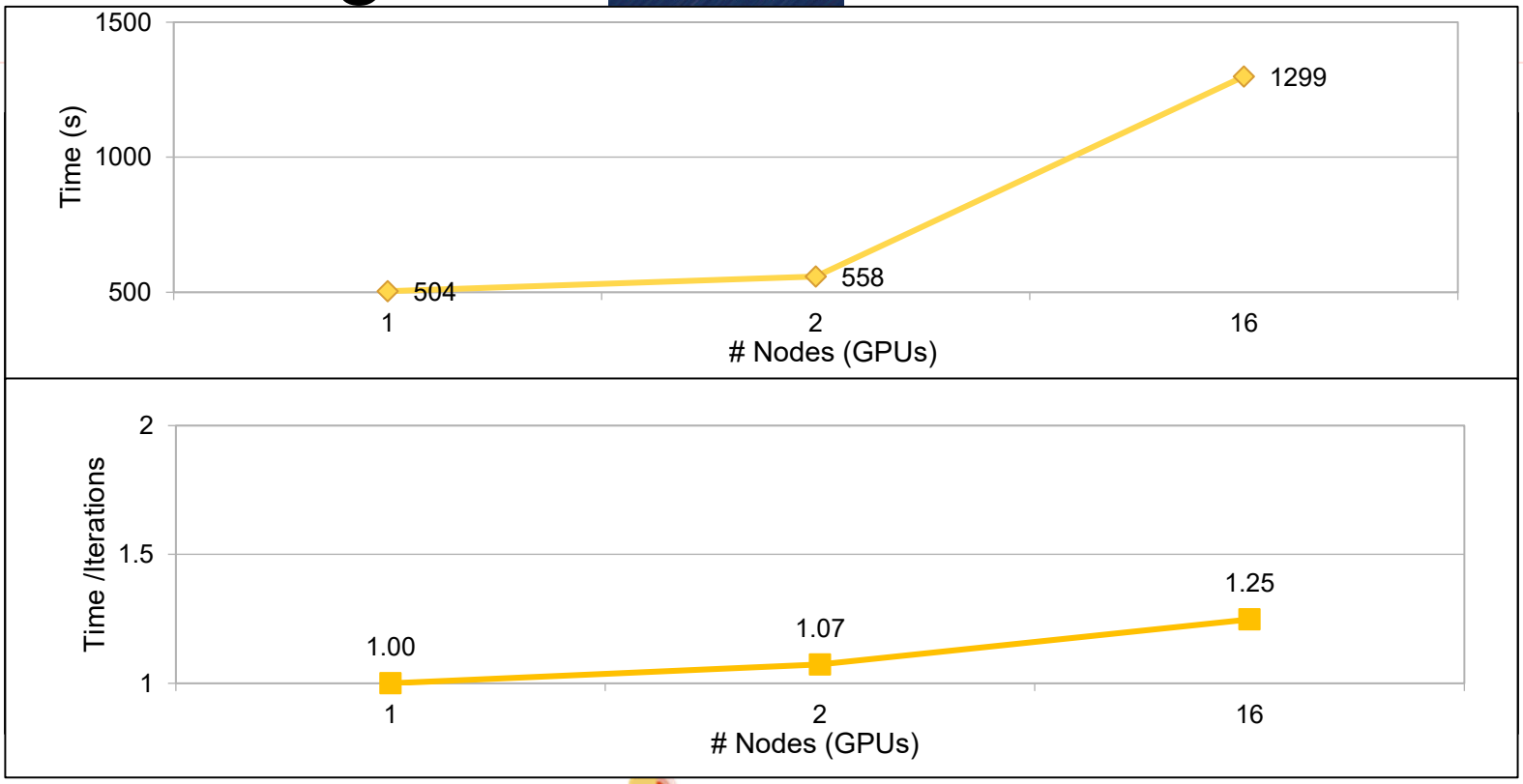
- @Marconi100
  2 x IBM Power9 8335-GTH + 4 x GPU NVIDIA V100 per node
  1 – 16 nodes

Strong Scaling



Weak Scaling

MPI + OpenACC More Science, Less Programming (CUDA-Aware MPI)



Outline

- FALL3D physics
- FALL3D components
- Implementation
  - Profiling
- FALL3D to fall3d
- ACCelerating
- Results
- **Conclusions**
- Future work

Conclusions

- The Mini-app FALL3D only solves task_fall3d assuming a fixed problem (helicoidal velocity field)
- OpenACC More Science, Less Programming is used to speedup both the Mini-app & the full-app of FALL3D
- **The current Mini-app GPUs are computing 100%** (vs ~27%)
- The Minifall3d app was successfully tested on both strong & *weak* scaling
- The latest Minifall3d GPU version (MPI + OpenACC) is ~ 7x faster than the previous version (MPI + OpenMP + OpenACC) on a medium size problem (400x400x240 grid cells)
- A speed-up of ~ 29x has achieved vs plain MPI version (after 2 NVIDIA hackathons)

Outline

- FALL3D physics
- FALL3D components
- Implementation
  - Profiling
- FALL3D to fall3d
- ACCelerating
- Results
- Conclusions
- **Future work**

Future work

- Merge fall3d into FALL3D
- Next GPU HACKATHONS NVIDIA

Questions?

Thank you very much

eduardo.cabrera@bsc.es