GENEX – GVEC - SPICE2d – ALYA

Alejandro Soba, Carlos Romero Madrid, Joan Vinyals

HPC ACHs | 25-26 November 2025

This work has been carried out within the framework of the EUROfusion Consortium, funded by the European Union via the Euratom Research and Training Programme (Grant Agreement No 101052200 — EUROfusion). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them.

GENE-X

Gyrokinetic code to simulate plasma turbulence in X-point geometries in tokamaks and stellarators.

Work required into ACH: Assessment of reordering algorithm using a multigrid approach to increase cache efficiency and reduce simulation time.

End of 2024 + complications with medical leave of CR:

- **Permissions revoked to Git** repositories and sub-repositories.
- **Compilation** via CMake requiring **internet access unavailable** at first stages of MareNostrum 5.
- Fails in the **large size** grids cases

3 PM passed to 2025: i) problems in GIT and Cmake in MN5 and Leonardo were fixed. ii) Errors were found in the several field solvers such as solve_maxwells_equations and solve_ohms_law within the timestep subroutine due to incorrect application of the reordering algorithm.

Final Work and Conclusions

After a developer's revision, a corrected version of the code was tested. New insights on reordering values will guide experiments with higher reordering levels. Previously, due to convergence issues, the timestep dt had to be reduced to **5e-7** due to solver errors. With the new fixed code, the value of the timestep has been restored to **1e-4.**

If we **discard the initialization time** and only consider the evolution, we are left with the following results in the `timestep` subroutine:

- **Small grids: 4–8%** reduction (varies with multigrid levels).
- **Medium grids: ~10%** reduction (only one multigrid level tested).
- **Large grids: ~10%** reduction across multiple reordering strategies (consistent with different multigrid levels).

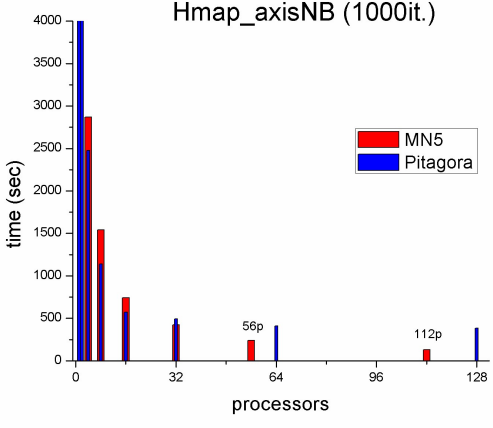
GVEC

3D MHD equilibrium code GVEC. Written in Fortran 90 + hdf5.

Work required into ACH: Hybrid parallel performance (OpenMPI + MPI). Identification of bottlenecks. Load Imbalance.

Task done in 2025

- Installed in MN5 and PITAGORA. Performance analysis (Openmp+MPI)
- EXTRAE/PARAVER analysis.
- DLB (TALP) Integration



Conclusions

- After the understanding of the different zones of the code and the meaning of this two regions, several improvements in the communications patterns can be do it.
- OPENMPI combined with MPI (2 or 4 processors) can be a possible paradigm to use to improve efficiency.
- Gfortran and Intel compilations were compared.
- The load balance analysis is presented by other integrant of our team.

SPICE2

- **SPICE** (SheathParticle In CEll) package includes two codes: SPICE2 (2D3V) and SPICE3 (3D3V).
- PIC code for simulations of particles in a fixed magnetic and self-consistent electric field
- Written in Fortran 90, outputs in the Matlab MAT binary format.
- Parallelization implemented using domain decomposition principles and message passing interface (MPI).
- All internal routines are parallel except for the Poisson solver. The Poisson solver is serial (taking 3% of the overall calculation time). It operates with global matrices of potential and charge density.
- Best solver are SPARCE, based in UMFPACK.

SPICE2

Initial Work required into ACH

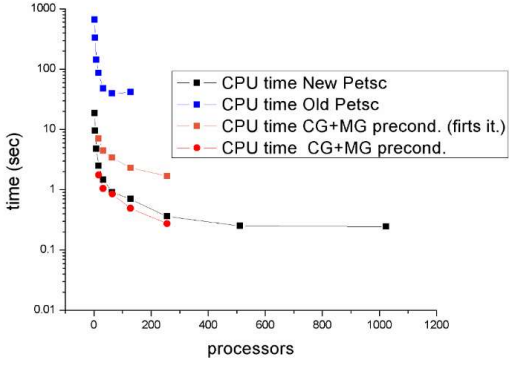
- Implementation of 2D parallel Poisson solver with good scaling and speed so that the number of cores in simulations can be increased to at least 128 (current practical limit is around 32) and the grid size can be increased (UMFPACK has a limit of ~4000 cells in one dimension);

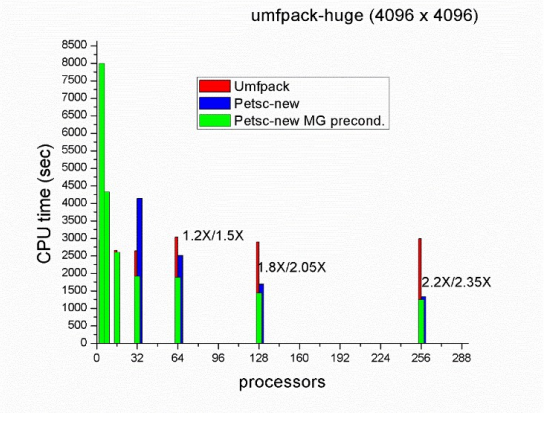
- Implementation of parallel routine for E-field calculation (later task).

New requirements for 2024/2025

- **Continue the solver development;**
- **Create a git branch for BSC version;**
- **Create parallel Outputs;**
- Reduce the memory consumption;
- Go to a complete parallel SPICE2d.

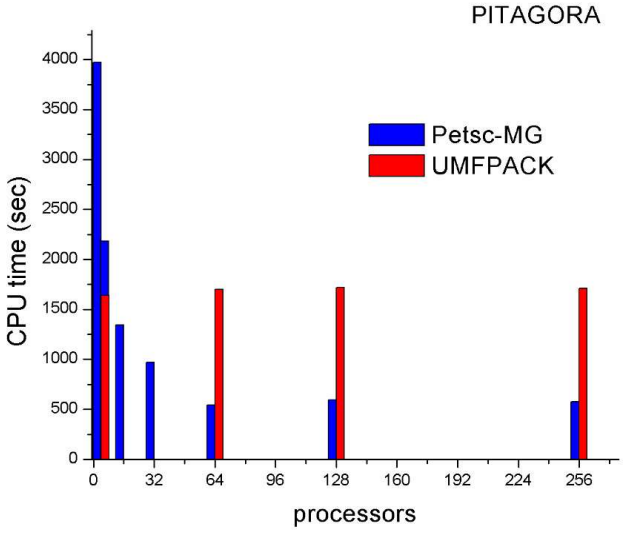
SPICE2





*Petsc computing time comparison **Algebraic Multigrid (AMG) Preconditioners** using the CPU time and iteration in the new version of the Petsc solver*

SPICE2



Case umfpack-huge.inp in Pitagora: After 64 processor we reach better CPU times than Umfpack.

SPICE2

The manage of memory (static) in SPICE2 derive in several problems. For big cases, the amounts of processors that can be used by node are small. By other side, a lot of time is consumed in read/write instances.

Proposed solution:

i) Past all the arrays to dynamic memory allocation. (Delete unused arrays)

ii) Turn on only the arrays used in each case according to the input variables.

iii) Use domain decomposition in all the “natural” possibilities

iv) Possible decomposition in Nz_extended (need developers’ help)

v) Diagnostic arrays are possible optimization (need developers' decision)

vi) Full domain used in all the processors (just 10% of the arrays) need to be analyzed.

SPICE2

Conclusions

- New preconditioner in Petsc generated a better performance.
- We fix the 60% of the arrays with bad distribution of the domain. This imply a 100% of efficiency improvement.
- For the largest benchmark case (4096×4096), the updated version executed 64 instances per node on MN5, resulting in higher efficiency and reduced execution time.
- Still remains the diagnostics arrays and the full used arrays analysis cases, but this need intervention of developers.

ALYA

Alya is a finite element framework created and actively developed by the Physical and numerical Modelling group within the Computer Applications in Science and Engineering (CASE) department of BSC. It implements MPI parallelization and domain subdivision in its kernel to leverage using several supercomputer nodes simultaneously.

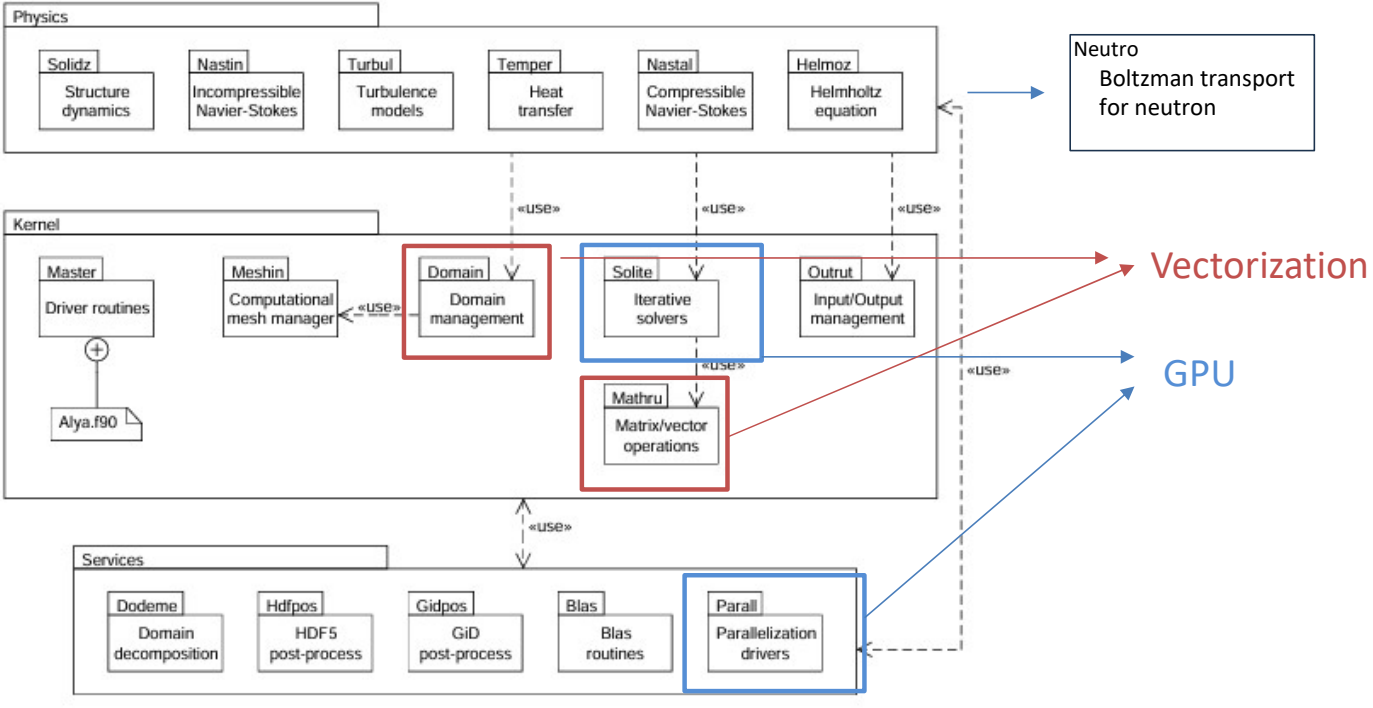
Code developers' requirements:

-) Perform optimizations in the KERNEL.
-) Vectorization of critical parts of the code.
-) Code refactoring leading to more efficient execution.

Big and complex code. We take the more demanding portion.

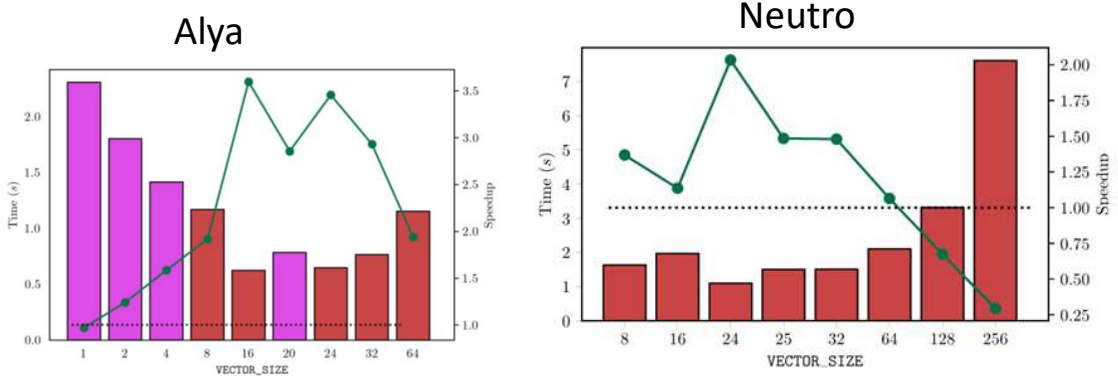
-) Combining shared memory schemes (OpenMP) with existing MPI implementation.
-) Porting solvers used in fusion modules to GPU (remains optimization).

ALYA


Physics
Solidz
Structure dynamics
Nastin
Incompressible Navier-Stokes
Turbul
Turbulence models
Temper
Heat transfer
Nastal
Compressible Navier-Stokes
Helmoz
Helmholtz equation
Neutro
Boltzman transport for neutron
«use»
«use»
«use»
Kernel
Master
Driver routines
Alya.f90
Meshin
Computational mesh manager
«use»
Domain
Domain management
Solite
Iterative solvers
Outrut
Input/Output management
«use»
«use»
Mathru
Matrix/vector operations
Vectorization
GPU
«use»
Services
Dodeme
Domain decomposition
Hdfpos
HDF5 post-process
Gidpos
GiD post-process
Blas
Blas routines
Parall
Parallelization drivers

Alya

Vectorization: It's necessary after the porting to GPU. We test the code with different election of the variable VECTOR_SIZE. (Default=16)



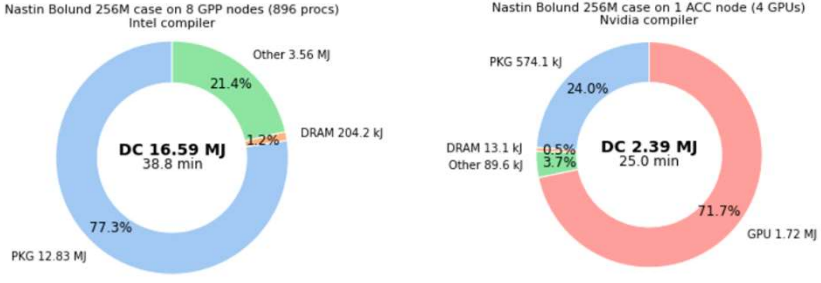
-) x3.6 speedup
-) x1.5 without autovect.
-) Compiler may need directives

Total runtime (bars) and speedup (line) with respect the non-vectorized version, for different values of VECTOR_SIZE.

Alya

GPU porting: Has two main tasks (with a lot of subtasks!)

1) Porting the kernel to GPU (Solvers, meshing, etc.)
2) Porting modules (NEUTRO, NASTIN, etc) to GPU:Implementing vectorized assembly routines that optimize memory usage patterns for both CPU and GPU execution environments. This approach eliminates heap memory constraints while maintaining parallel efficiency across different hardware architectures



Some measures comparing CPU (8nodes, 896proccesors) with GPU (1node, 4GPU). Not only is faster, is less energetic consumer

General Conclusions

- The required work with GENE-X were finished and all the reordering test were done successfully (problems fixed!)
- We complete the requirements for GVEC code, and we discuss with the developers several possible improvements. He decide to ask time to fix other urgent issues in MPG.
- The work with SPICE2 is close to reach a final point. The requirements of developers were done and the recommendation of ACH-BSC almost done (the final decision need to be taken by the developers.)
- Several improvements were done in Alya code, but to do all the work, more time is needed!. Specially in the GPU porting (of more solvers) and optimization of this porting. Refactorizing of the code is delayed to next requests.

GENEX – GVEC - SPICE2d – ALYA

Many Thanks!

HPC ACHs | 25-26 November 2025

This work has been carried out within the framework of the EUROfusion Consortium, funded by the European Union via the Euratom Research and Training Programme (Grant Agreement No 101052200 — EUROfusion). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them.