PARALLEL COMPUTATION
FOR THE ADAPT FRAMEWORK

Fayssal Benkhaldoun*, Christophe Cerin[t], Imad Kissami[t]

* LAGA, Université Paris 13, 99 Av J.B. Clement, 93430 Villetaneuse, France

[t] LIPN, Université Paris 13, 99 Av J.B. Clement, 93430 Villetaneuse, France

Introduction

In order to run Computational Fluid Dynamics (CFD) codes on large scale infrastructures, parallel computing must be used because of the computational intensive nature of the problems.

Goals

We investigate the ADAPT framework and we deal with the parallel multi-frontal direct solver (MUMPS) and mesh partitioning methods using METIS to improve the performance of the framework, because the 3D Streamer code takes up to 30 days to give results.

Streamer equation

$$\frac{\partial n_e}{\partial t} + div(n_e \vec{v_e} - De\vec{\nabla} n_e) = S_e, \quad (1)$$

$$\frac{\partial n_i}{\partial t} = S^+, \quad (2)$$

$$\Delta V = -\frac{e}{\varepsilon}(n_i - n_e)\ , \quad (3)$$

$$\vec{E} = -\vec{\nabla} V\ , \quad (4)$$

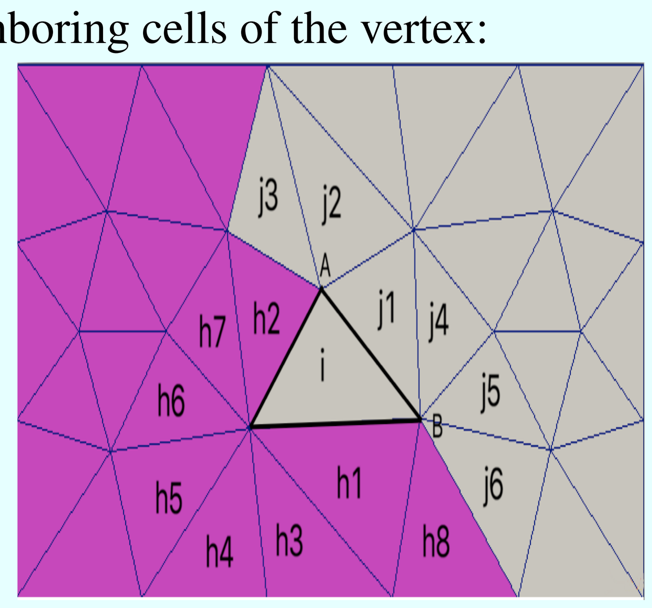
The third equation leads to a system of linear equation :

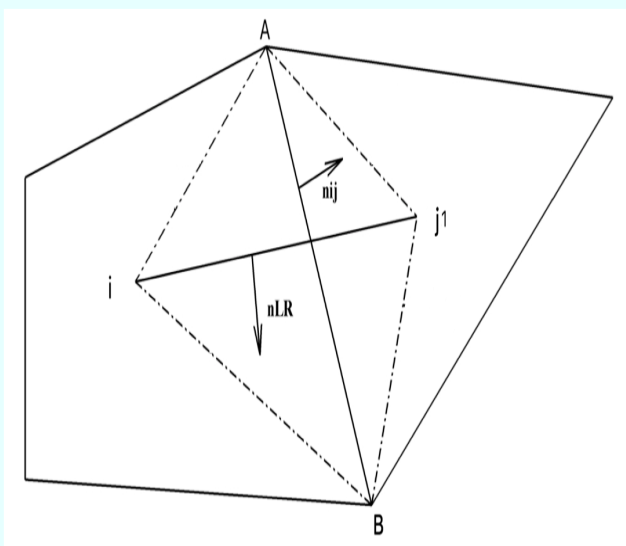
$$A.\vec{V}^n = Q^n$$

This system is solved directly by LU decomposition with an implementation for sparse matrices (we use MUMPS solver).
The Numerical method requires the approximation of the solution Gradient on Cell boundaries:

$$\vec{\nabla} n_{eij} = \frac{1}{2\mu(D_{\sigma_{ij}})}[(n_e(A) - n_e(B))\vec{n}_{LR}|\sigma_{LR}| + (n_e(j) - n_e(i))\vec{n}_{ij}|\sigma_{ij}|] \quad (5)$$

$n_e(A)$ and $n_e(B)$ are estimated by a local least-square projection using the values of the neighboring cells of the vertex:





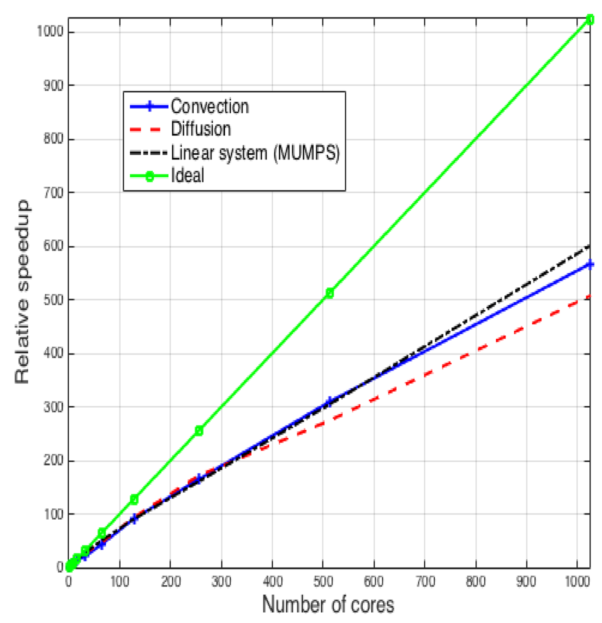
Parallel workflow of ADAPT :

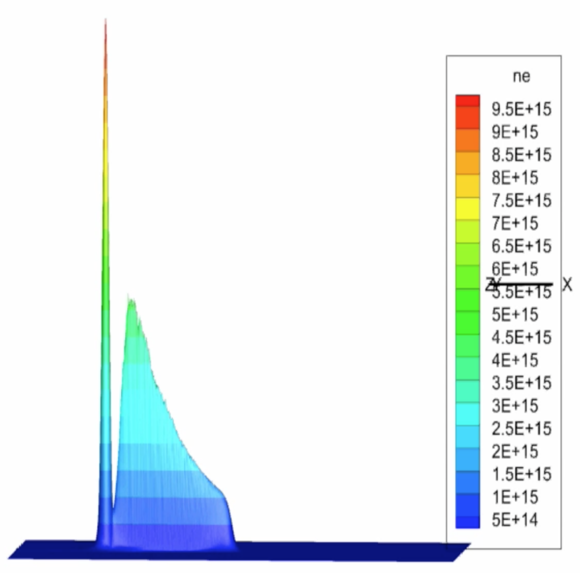
- Cutting 2D and 3D meshes using METIS,
- Construction of halo cells (cells which are at the boundary of sub-domain),
- Using MUMPS to solve the lineair system,
- Save results in parallel way using Paraview.

Parallel 2D Streamer results :

Compute cores	Total	Convection	Diffusion	Linear solver
1	49 h 54 min 48 s	02 h 51 min 04 s	13 h 06 min 00 s	33 h 57 min 44 s
2	25 h 06 min 27 s	01 h 22 min 57 s	06 h 41 min 02 s	17 h 02 min 27 s
8	06 h 27 min 18 s	00 h 22 min 13 s	01 h 46 min 26 s	04 h 18 min 38 s
64	01 h 01 min 41 s	00 h 03 min 59 s	00 h 17 min 05 s	00 h 40 min 36 s
256	00 h 18 min 16 s	00 h 01 min 02 s	00 h 04 min 34 s	00 h 12 min 39 s
1024	00 h 05 min 14 s	00 h 00 min 18 s	00 h 01 min 33 s	00 h 03 min 23 s

Execution time of different parts of parallel 2D streamer code using mesh with 529240 cells



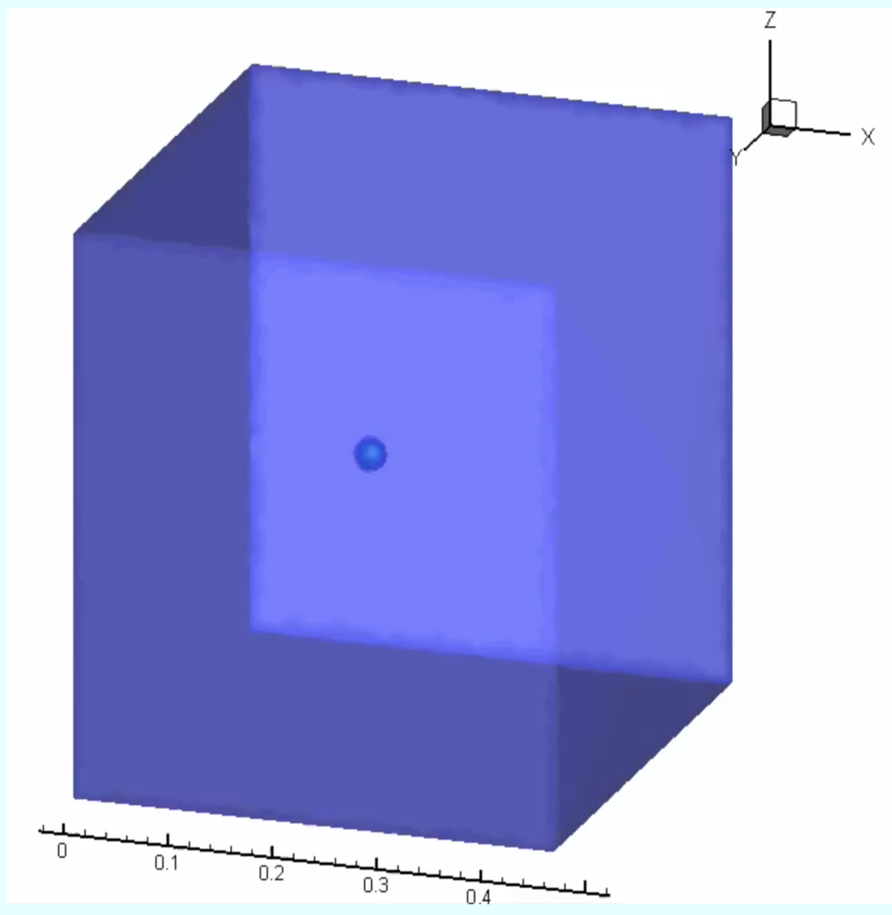


Speedup of different parts of 2D streamer code (left) and 2D Streamer propagation (right)

Parallel 3D transport coupled with Parallel solver results :

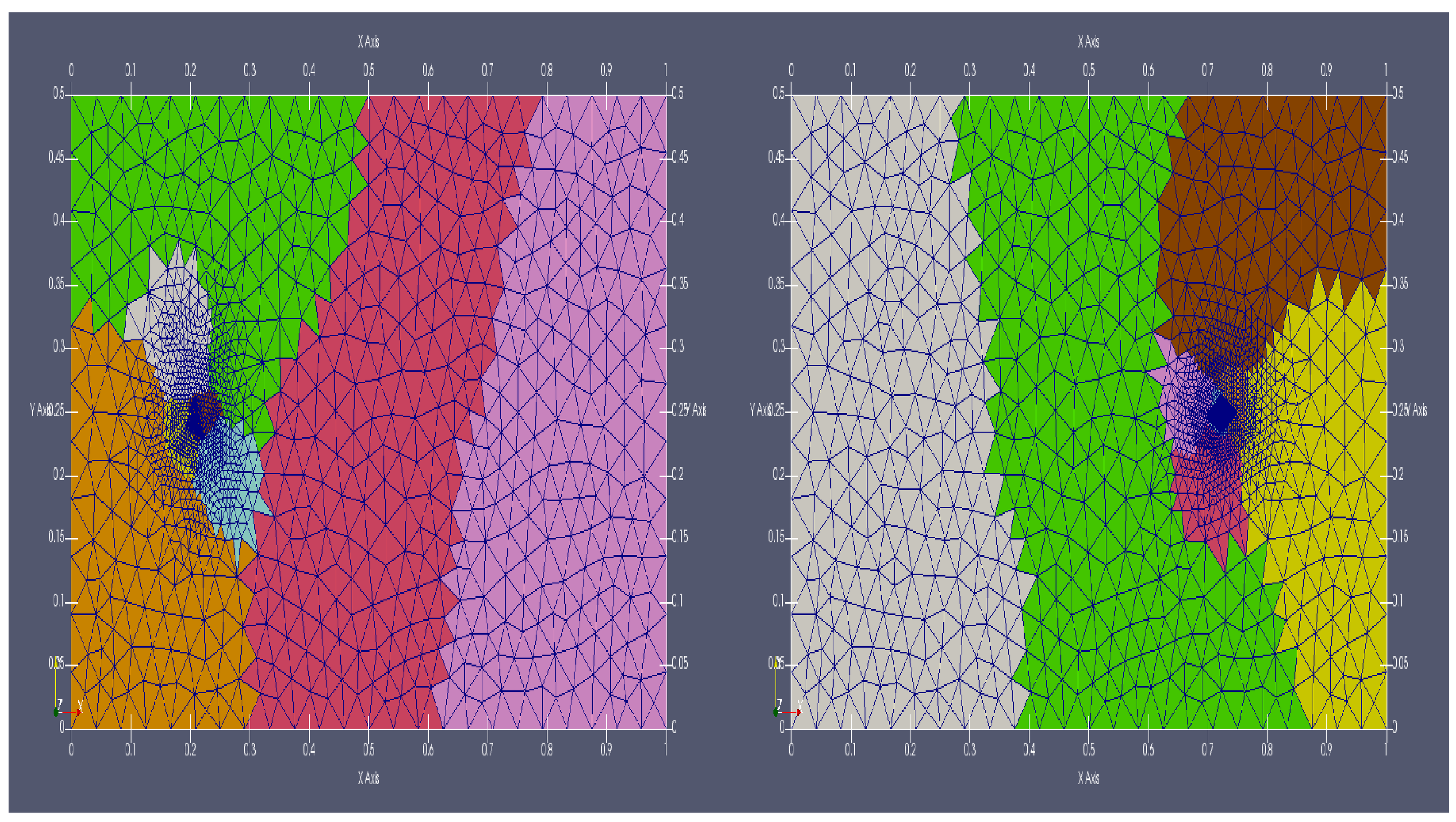
Compute cores	Execution time	Speedup
1	262 h 22 min 11 s	1
2	139 h 33 min 31 s	1.88
4	77 h 30 min 34 s	3.38
8	42 h 50 min 10 s	6.12
16	24 h 57 min 25 s	10.51
32	14 h 30 min 20 s	18.08
64	07 h 52 min 16 s	33.32
128	04 h 20 min 59 s	60.30
256	02 h 29 min 06 s	105.50

Execution time of parallel 3D code using mesh with 557542 cells



3D transport propagation

Parallel 2D transport using dynamic Adaptatif mesh :



Conclusion

Contributions :

- Mesh partitioning using METIS.
- Parallelization of the linear system using MUMPS solver.
- Parallelization of the 2D Streamer code which considers simultaneously the parallelization of the linear system and the evolution equation.
- Parallelization of 3D Streamer.

Future work :

- The challenge is to mix mesh partitioning and dynamic mesh adaptations.