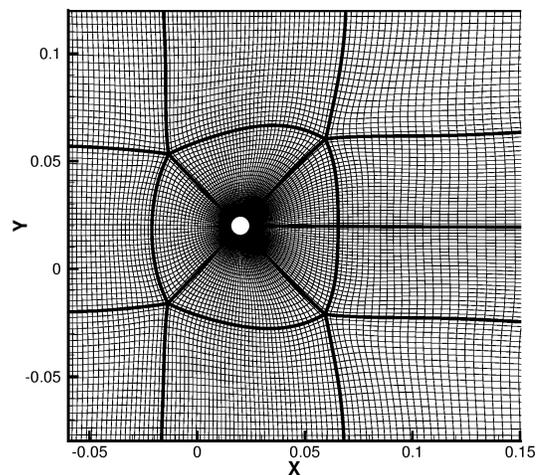


Optimal partitioning of multi-block structured grids

Patrice Castonguay¹

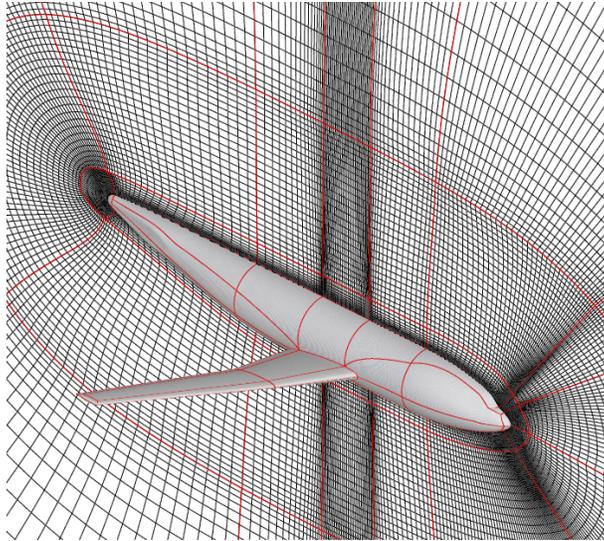
Bombardier Aerospace, Montreal

One heavily used CFD code at Bombardier is called Fansc, which solves the Navier-Stokes equations on so-called multi-block structured grids. A multi-block structured grid contains an unstructured arrangement of hexahedral blocks, where each block contains a structured grid. An instance of such a grid (in 2D for simplicity) is shown below.

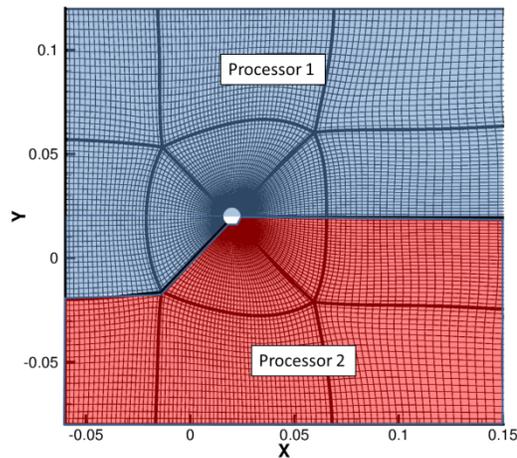


Below is another instance on an aircraft geometry (the edges of the blocks on the surfaces are shown in red). Typically the grids we use contain on the order of 40-100 blocks and a total of 20 million cells.

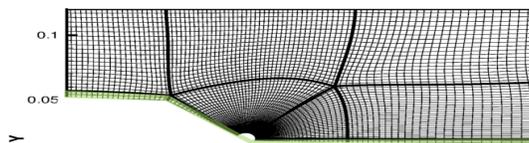
¹ Advanced Aerodynamics Dept., Tel. 514-855-5001 x.12345, patrice.castonguay@aero.bombardier.com

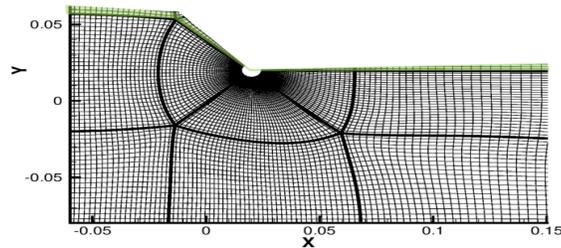


To speed up computations on such grids, multiple processing units (cores) are used, where each processing unit is allocated the task of computing the solution on a portion of the grid. In Fansc, this is done by assigning blocks to different processing units. An example is shown below, where Processor 1 is allocated the task of computing the solution on the 8 blue blocks, while processor 2 is allocated the task of computing the solution on the 6 red blocks.

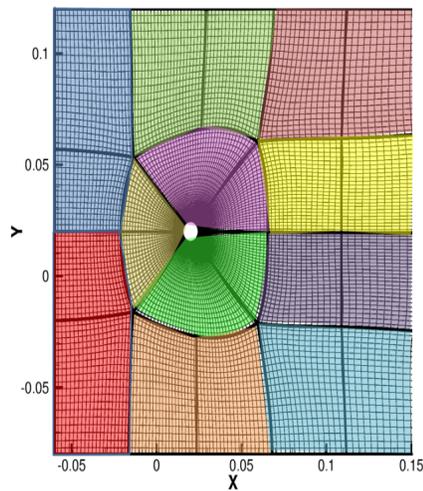


Since those processing units are operating in parallel, they must communicate frequently during the solution process to exchange information at the boundary of their respective computational domains. In Fansc, this is done by exchanging information in two layers of cells (called halo cells) between processing units at regular intervals during the simulation. In the mesh below the halo cells are shown in green.





For the parallel computation to scale well, the numbers of cells in any two processors must be similar (this property is called “good load balance”) and communication between processors must be minimized. In general the number of blocks in the grid is independent of the number of processing units and blocks may have different numbers of cells; thus we may consider splitting blocks in order to achieve good load balance and minimize communication. In the example below, some blocks have been split to achieve good load balance for a simulation with 11 processing units.



When the number of blocks is large, it is not trivial to devise the optimal way of splitting blocks and assigning them to the processing units. Furthermore we want to keep the blocks as large as possible because this is more efficient from a computational point of view.

Simple heuristic block-splitting and load-balancing algorithms were developed in [1]. While these algorithms provide a reasonable parallel speed-up for the typical grid sizes and computational resource allocations (i.e., number of processors) used so far, they do not attempt to minimize the communication or halo cell overhead.

We are therefore interested in developing an algorithm to split blocks (and maybe recombine them) and assign them to processing units in order to:

- 1) Achieve good load balance;
- 2) Minimize communication between processing units; and
- 3) Keep the blocks as large as possible

Reference

- [1] K. Sermeus, E. Laurendeau, F. Parpia, "Parallelization and performance optimization of Bombardier multiblock structured Navier-Stokes solver on IBM eserver cluster 1600," AIAA Paper 2007-1109.