Joris Degroote

Solving time-dependent multi-physics problems using multiple instances of the same solvers

Ghent University
Department of Flow
Heat and Combustion Mechanics
Sint-Pietersnieuwstraat 41
B-9000 Ghent
Belgium
Joris.Degroote@UGent.be
Robby Haelterman
Jan Vierendeels

This research aims at accelerating the simulation of fluid-structure interaction (FSI), which is a multi-physics problem involving the mutual influence between a fluid flow and a deformable structure. In addition to the governing equations in the fluid and structure domains, also two equilibrium conditions have to be satisfied on the contact surface between these domains, namely equality of velocity and traction on both sides of the contact surface. Examples of fluid-structure interaction problems are the motion of heart valves, the vibration of heat exchanger tubes and the bending of wind turbine blades.

Fluid-structure interaction problems can be solved in a monolithic or a partitioned way. The monolithic approach is to solve all equations simultaneously, taking into account the interaction between the fluid and the structure during the solution process. Conversely, the partitioned approach is to solve the equations for the fluid and for the structure separately and to perform coupling iterations until the equilibrium conditions are satisfied. The main advantage of the partitioned approach is that it allows to reuse existing solvers for the fluid flow and for the structural deformation. In this research, the focus lies on partitioned techniques and more specifically on quasi-Newton methods for time-dependent simulations.

Traditionally, only one flow solver and one structural solver are used for a fluid-structure interaction simulation. Each of them can run on multiple cores in a parallel calculation in which the domain is split over the different cores. However, there is typically a limit to the number of cores which can be used to accelerate the simulation, depending on the size of the problem, the computer hardware and the parallelization of the software. Therefore, the main idea of this research is to use multiple flow solvers and multiple structural solvers, all solving the same nonlinear problem and each one of them possibly running on multiple cores. Hence, an additional level of parallelization is obtained, besides

the domain decomposition.

The presented algorithm considers one of the flow solvers and one of the structural solvers as the masters, which actually calculate the solution of the nonlinear problem in the current time step using a quasi-Newton algorithm. Obviously, the convergence of the quasi-Newton iterations is faster if the approximation for the Jacobian is better. In the quasi-Newton methods, this approximation is constructed using the information obtained during the quasi-Newton steps, by considering differences between consecutive iterations. The additional solvers calculate the residual of the nonlinear problem in the current time step after adding a step from the previous time step to the current value of the solution. The resulting information in the current time step helps to improve to approximation of the Jacobian and consequently accelerates the coupling iterations between the master solvers. Clearly, this is only useful for time-dependent problems.

As a test case, the propagation of a pressure wave through an incompressible and inviscid fluid in a flexible tube is calculated. As the number of flow solvers and structural solvers is increased from 1 to 8, the average number of coupling iterations per time step reduces from approximately 8 to 3. Consequently, a speed up of the calculation with a factor more than 2 is obtained, on top of the speed up due to parallelization by domain decomposition. The results indicate a relatively low parallel efficiency for this additional level of parallelization, so increasing the number of solvers should only be applied if it results in a higher speed up than increasing the number of cores per solver. Furthermore, it has to be noted that the exact number of coupling iterations depends on the system load as the additional solvers can finish their calculation slightly before or after the master solvers perform a quasi-Newton step, resulting in the inclusion of additional information in the approximate Jacobian or not.

This so-called multi-solver quasi-Newton algorithm is only considered as an initial step in the development of this kind of algorithms and hopefully also other algorithms with multiple solvers will be developed.