The Eleventh International Conference on
Engineering Computational Technology
23–25 August, 2022 | Montpellier, France

Proceedings of the Eleventh International Conference on
Engineering Computational Technology
Edited by B.H.V. Topping and P. Iványi
Civil-Comp Conferences, Volume 2, Paper 2.1
Civil-Comp Press, Edinburgh, United Kingdom, 2022, doi: 10.4203/ccc.2.2.1

On the practical parallel implementation of a monolithic fluid-structure interaction solver

M.-H. Chen[1], P.K. Jimack[2] and Y. Wang[2]

[1]Department of Mathematics, National Chung Cheng University, Taiwan
[2]School of Computing, University of Leeds, United Kingdom

Abstract

This paper describes the parallel implementation of a monolithic fluid-structure interaction (FSI) algorithm that has recently been proposed as a robust solver for the deformation of both soft and stiff solid structures interacting with an incompressible fluid. The FSI solver is based upon a modification of the Navier-Stokes operator, to account for the presence of the solid structure in certain regions of the domain. Consequently, the parallel solver that is proposed is related to the domain decomposition method for finite element discretizations of incompressible flows. In particular, a partial assembly of the finite element system is undertaken so as to avoid communication between processors at this stage: however, when an iterative solver is applied neighbour-to-neighbour communication is required at each iteration. Furthermore, we employ a parallel preconditioner that is also based upon the domain decomposition method, with a suitable adaptation to account for the presence of the solid structure. Results are presented to illustrate the strong scaling performance on a representative test problem and the application to further problems is discussed.

Keywords: fluid-structure interactions, incompressible flow, parallel algorithms, finite element method, MinRES.

1 Introduction

We describe the parallel implementation of a monolithic fluid-structure interaction (FSI) solver that is based upon the *one-field fictitious domain method*, introduced in [1]. In this approach the structure (assumed to be a hyper-elastic solid material) is

discretized using an unstructured finite element representation, which overlays a single mesh for the whole of the computational domain (occupied by both the fluid and the structure). The governing equations are then assembled on the single mesh, based upon the Navier-Stokes equations for the fluid with a suitable modification to reflect the presence of the solid structure. The main dependent variable is the fluid velocity everywhere, including in the region occupied by the structure (hence the term *fictitious domain* (FD)). In this region the velocity is interpolated onto the mesh of the structure and used to deform it prior to the next time step.

When a mixed finite element (FE) method is used to discretize the incompressible Navier-Stokes (NS) equations the resulting nonlinear system at each implicit time step may be split into a convection half-step followed by a diffusion half step [2]. The former may be solved using the Taylor-Galerkin method, for example, whilst the diffusion problem gives rise to a saddle point system of the form:

$$\begin{pmatrix} A & B \\ B^T & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}$$

Here, u and p are the vectors of unknown velocity and pressure coefficients at the corresponding FE nodes; $A=M/dt+K$ (where M is the FE mass matrix, arising from the $d\underline{u}/dt$ term in the NS equations, and K is the stiffness matrix, arising from the $div(\sigma^f)$ term); and B is a matrix arising from the discretization of the *grad(p)* term in the NS equations, [3].

In our FD method, the modification to account for the presence of the structure has no effect on the convection half-step but changes the expressions for A and b in the saddle point system. Specifically, $A=M/dt+K+D^T(M^S/dt+K^S)D$, where: D is the interpolation matrix between the mesh covering the whole domain and the mesh of the structure; M^S is the mass matrix for the structure; and K^S is the stiffness matrix for the structure (arising from the $div(\sigma^s)$ term in the nonlinear elasticity equations). The purpose of this work is to develop a simple parallel solver for this modified saddle point system based upon its parallel assembly followed by a parallel preconditioned MinRES solver [4].

2 Methods

In this section we discuss the two main components to our parallel implementation of the *one-field fictitious domain method*: assembly and solution.

For the finite element discretization of the whole domain we use a block-structured mesh provided by PARAMESH [5]. This defines the mesh as the union of a number of mesh blocks, where each element of the mesh is a quadrilateral in 2D or a regular hexahedron in 3D. We use the Taylor-Hood finite element pair, [3], which consists of piecewise biquadratic (triquadratic) velocities and piecewise bilinear (trilinear) pressures in 2D (3D). The solid region is meshed using triangles in 2D (tetrahedra in 3D), with displacements/velocities represented by linear finite elements. The whole domain is partitioned in space within PARAMESH, with each block assigned to a process (and each process holding the data for multiple contiguous blocks: a subdomain). The FE assembly of the matrices M and K is undertaken independently on each block within PARAMESH, thus allowing for a high degree of parallelism. The assembly of the contributions of $D^T(M^S/dt+K^S)D$ on each subdomain is more challenging however, potentially requiring substantial communication and

computational overhead in order to identify the subdomains that are covered by any of the elements in the mesh of the solid structure. We avoid this additional complexity by holding a copy of the whole of the solid mesh on each process, and then computing the above contributions without the need for any communication overheads. Clearly this comes at the price of some unnecessary work on each process however.

The parallel MinRES solver that we have implemented does not require the full assembly of the matrix blocks *A* and *B*: instead communication between neighbouring blocks takes place at the steps in the algorithm that require inner products of global vectors and a matrix-vector product to be computed. This communication is facilitated within PARAMESH through the use of guard cells that surround each block. A further enhancement of the solver is the use of a preconditioner to improve the convergence of the MinRES solver. This is based upon an incomplete Cholesky decomposition (with zero fill-in) of the following block diagonal matrix (where M_p is the mass matrix for the pressure space):

$$\begin{pmatrix} M/dt + K & 0 \\ 0 & M_p/dt \end{pmatrix}$$

For the parallel implementation we omit the connections between each subdomain when computing this incomplete decomposition. Hence the effectiveness of the parallel preconditioner is expected to deteriorate as the number of parallel processes/subdomains is increased.

3 Results

For this short paper we focus on one representative numerical example in 2D: computing the transport and deformation of a soft solid disc in a lid-driven cavity flow, [6]. The overall domain is (0,1) x (0,1), with Dirichlet conditions imposed on the velocity field throughout the boundary: on three of the four sides the velocity is zero, whilst on the top boundary (the driving lid) it is $(0,1)^T$. Initially, a round stress-free disc of radius 0.2 is centred at (0.6,0.5), and the simulation proceeds on a 128 x 128 global mesh (with the solid represented by a mesh of 31163 triangles) using a time step of 0.01.

Figure 1 shows snapshots of the solution at four different times, 0.5 (shortly after the disc begins to deform), 2.0, 4.0 and 6.0, which match those computed by our sequential code in [1] (using the same material parameters cited therein). There is clearly very large deformation in the solid, which has been captured effectively by the solver. The parallel performance for this example is illustrated in Table 1. All simulations were carried out on *Taiwania*, a supercomputer having a memory of 3.4 petabytes and delivering over 1.33 quadrillion flop/s of theoretical peak performance. The system has 630 compute nodes based on 40 core Intel Xeon Gold 6148 processors running at 2.4 GHz. By using our parallel implementation we are able to reduce the run time for a typical simulation from ~65 hours (for 800 time steps) to under 90 minutes.

Although there is insufficient space to include full results here, when we solve a corresponding problem in 3D even greater speed-ups are possible: reducing a sequential simulation from over 13 days to approximately 3 hours on 512 cores. As

will be discussed further in the next section, there are two main contributory factors to the degradation in the strong scaling as the number of processors is increased.

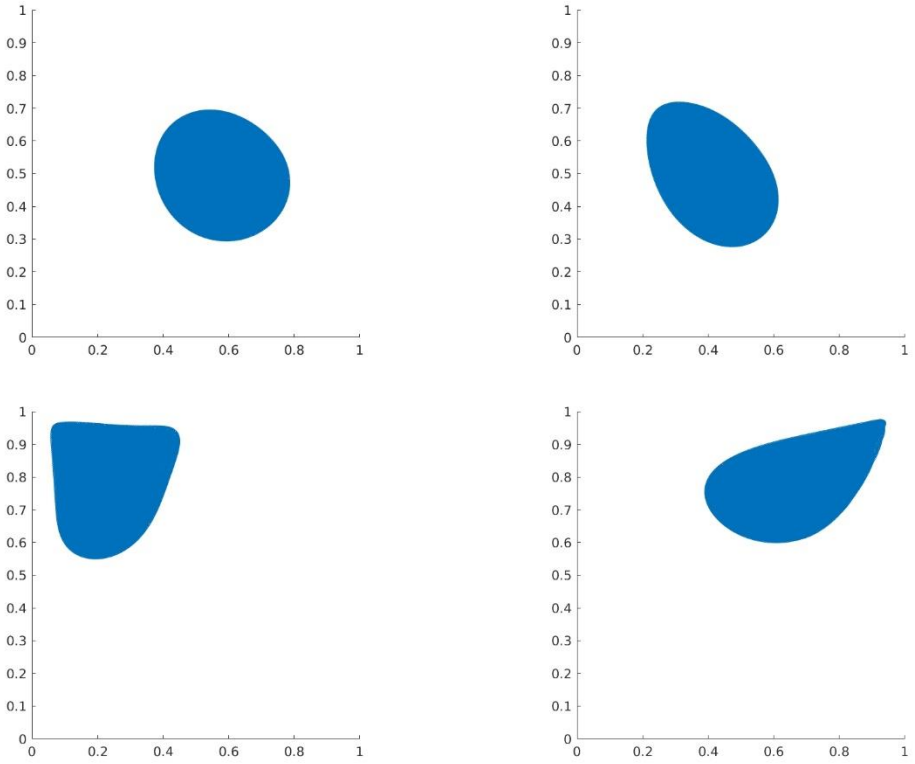

Figure 1: Simulation results showing the deformed disc at t=0.5 (top left), 2.0 (top right), 4.0 (bottom left) and 6.0 (bottom right).

For the final 100 time steps	Serial	P=4	P=16	P=64	P=256
Wall clock time (minutes)	490	148	48	23	11
Speed-up	1	3.3	10.2	21.3	44.5
Average MinRES iterations per step	20.3	24.2	27.5	29.8	31.1

Table 1: Parallel performance for the test case consisting of a 2D deforming solid in a lid-driven cavity flow.

4 Conclusions and Contributions

In this work we have developed and tested the first parallel implementation of the *One-field fictitious domain method*, for solving general fluid-structure interaction (FSI) problems. This method has been shown to have desirable stability properties, [7], and to be highly effective in addressing a wide range of FSI applications, [1,8]. However, like other monolithic approaches to FSI problems (e.g. [9,10]), this FD method incurs a large computational cost at each implicit time step, particularly for

the diffusion part which requires the solution of a large sparse saddle-point system. We have developed a parallel implementation of this solver which is based upon partial assembly of the relevant finite element matrices (independently assembled on each subdomain), a parallel preconditioner, and a parallel MinRES implementation.

The strong scaling on a 2D test problem is shown to yield a speed-up of 44.5 on 256 cores, thus reducing run times of two days to little more than an hour. Nevertheless, the efficiency does deteriorate notably as the number of processes, P, increases. One contribution to this is the fact that the incomplete factorization that is used as a preconditioner becomes more and more sparse as P grows (since, to avoid additional communications, we undertake the factorization process independently on each subdomain). In the case were P=256 this leads to more than a 50% increase in the number of MinRES iterations required at each time step: hence the speedup per iteration is in fact equal to 68.2 on 256 cores. The second major contributor to loss of efficiency with growing P is the decision to store a copy of the solid mesh on each process. This saves a large amount of searching and bookkeeping in order to identify which subdomains the triangulation intersects, and is also helpful with load-balancing, however it does impose a fixed overhead per processor. Consequently, by simply application of Amdahl's law, this places a clear limit on the degree of parallelization that will be possible for a problem of fixed size.

In conclusion, we have developed a new parallel implementation of a recently proposed algorithm for general FSI problems. For fixed problem sizes it has been demonstrated to allow speedups of close to two orders of magnitude on up to 512 cores. Its greatest potential however comes through weak scaling, where increasing the problem size (e.g. the finite element mesh resolution) in proportion to the number of cores is likely to show enhanced parallel efficiency.

Acknowledgements

The authors are grateful for access to the *Tiawania* supercomputer.

References

[1] Y. Wang, P.K. Jimack, M. A. Walkley. "A one-field monolithic fictitious domain method for fluid-structure interactions", Computer Methods in Applied Mechanics and Engineering, 317, 1146–1168, 2017.

[2] O. Zienkiewic, "The finite element method for fluid dynamics", 6th Edition, Elsevier BV., 2005.

[3] H.C. Elman, D.J. Silvester, A.J. Wathen, "Finite elements and fast iterative solvers", Oxford University Press, 2005.

[4] C.C. Paige, M.A. Saunders, "Solution of sparse indefinite systems of linear equations", SIAM J. Numer. Anal., 12, 617-629, 1975.

[5] K. Olson, P. MacNeice, "An Overview of the PARAMESH AMR Software and Some of Its Applications", in T. Plewa, T. Linde, G. Weirs (eds.), "Adaptive mesh refinement - theory and applications", Lecture Notes in Computational Science and Engineering 41, Springer, 2005.

[6] H. Zhao, J.B. Freund, R.D. Moser, “A fixed-mesh method for incompressible flow structure systems with finite solid deformations”, Journal of Computational Physics, 227, 3114-3140, 2008.
[7] Y. Wang, P.K. Jimack, M.A. Walkley, “Energy analysis for the one-field fictitious domain method for fluid-structure interactions”, Applied Numerical Mathematics, 140, 165–182, 2019.
[8] Y. Wang, P.K. Jimack, M.A. Walkley, “A theoretical and numerical investigation of a family of immersed finite element methods”, Journal of Fluids and Structures, 91, 102754, 2019.
[9] M. Heil, “An efficient solver for the fully coupled solution of large displacement fluid-structure interaction problems”, Computer Methods in Applied Mechanics and Engineering, 193: 1-23, 2004.
[10] C.-Y. Chiang, O. Pironneau, T.W. Sheu, M. Thiriet, “Numerical study of a 3d Eulerian monolithic formulation for incompressible fluid-structures systems”, Fluids, 2, 34, 2017.