Parallel simulation of three-dimensional free-surface fluid flow problems

T. A. Baer[1], S. R. Subia[2], & P. A. Sackinger[2]

[1] *Incompressible Fluid Mechanics Department*
email: tabaer@sandia.gov

[2] *Computational Thermal/Fluid Engineering Sciences*
Sandia National Laboratories
Albuquerque, New Mexico USA

Abstract

We describe parallel simulations of viscous, incompressible, free surface, Newtonian fluid flow problems that include dynamic contact lines. The Galerkin finite element method was used to discretize the fully-coupled governing conservation equations and a "pseudo-solid" mesh mapping approach was used to determine the shape of the free surface. In this approach, the finite element mesh is allowed to deform to satisfy quasi-static solid mechanics equations subject to geometric or kinematic constraints on the boundaries. As a result, nodal displacements must be included in the set of problem unknowns. Issues concerning the proper constraints along the solid-fluid dynamic contact line in three dimensions are discussed.

Parallel computations are carried out for an example taken from the coating flow industry, flow in the vicinity of a slot coater edge. This is a three-dimensional free-surface problem possessing a contact line that advances at the web speed in one region but transitions to static behavior in another part of the flow domain. Discussion focuses on parallel speedups for fixed problem size, a class of problems of immediate practical importance.

DISCLAIMER

1 Introduction

Many durable goods and electronic devices in use today exploit material features related to surface characteristics over parts of the product. Regardless of whether these surface characteristics are of a decorative or truly functional nature, the manufacture of these items begins with the application of thin liquid layers over a large surface area of substrate. The desired material features are obtained by controlling the shape and thickness of the layers to produce a optimum surface characterization during a subsequent curing or drying phase. Application of the fluid layers is often complicated by a number of factors affecting the free surfaces bounding the fluid layer that include fluid viscosity, gravity, surface tension, the liquid flow rate, the substrate motion, and curing or drying rates. Although these factors are well recognized, the techniques and procedures used in the manufacture of these goods still rely heavily upon experience rather than a purely analytical design approach. Realistically, an analytical design approach is often not feasible due to the large number of factors governing the fluid layer flow and the manner in which competing interactions affect the flow. In recent years there has been an increased interest in using a numerical modeling approach to develop and optimize new product designs. The approach used herein is to employ a finite element method for this class of numerical simulations. In particular, we employ the Sandia National Laboratories (SNL) finite-element computer code GOMA [1] for this purpose.

GOMA is a two-dimensional and three-dimensional finite element computer program incorporating original algorithms that make it especially suited for the analysis of certain manufacturing processes and free-surface flows. The program includes a full-Newton coupling of heat, mass, momentum, and pseudo-solid mesh motion algorithm that facilitates the simulation of processes in which the bulk fluid transport is strongly coupled to interfacial physics, even when the coupling is only implicit. Capabilities of the code are enhanced by its ability to treat not only fixed boundaries but also free and moving boundaries, including boundaries moving with specified kinematics. These features make it suitable to the study of problems in which the boundary position or boundary motion is a function of the problem physics, in geometrical design studies, and in fluid-structure interaction problems.

Numerical simulations of viscous three-dimensional fluid flow typically involve a large number of unknowns. When free surfaces are included, the number of unknowns increases dramatically. Consequently, this class of problem is a candidate for the application of parallel high-performance computing primarily because a larger number of compute nodes might allow one to solve larger problems. In the past, the GOMA code has been used primarily as a serial code but more recently we have undertaken the task of carrying out a parallel implementation of the code.

Issues affecting efficient parallel simulations include problem decomposition to equally distribute computational work among a SPMD computer and determination of robust, scalable preconditioners for the distributed matrix systems that must be solved. Solution continuation strategies important for serial simulations have an enhanced relevance in a parallel computing environment due to the difficulty of solving large scale systems.

This paper describes some issues surrounding the use of the parallelized GOMA code to analyze a nontrivial fluid flow problem. The model problem, an industrial slot coater flow including dynamic fluid/solid contact lines, is described. Some aspects of the solution procedure are presented along with a discussion on the appropriate boundary conditions for free-surface flow problems. Finally we provide timing results for parallel simulations of the slot coater flow for various numbers of processors. Parallel speedups for fixed problem size will be discussed.

2 Governing Equations

In addressing problems of coating flow we solve the equations for conservation of mass and conservation of momentum for the incompressible fluid

$$\nabla \cdot \mathbf{v} = 0 \tag{1}$$

$$\rho \frac{\partial \mathbf{v}}{\partial t} = -\rho(\mathbf{v} - \mathbf{v}_m) \cdot \nabla \mathbf{v} + \mathbf{g} + \nabla \cdot \mathbf{T} \tag{2}$$

along with the constitutive equation for a Newtonian fluid

$$\mathbf{T} = \mu \mathbf{D} - p\mathbf{I} \quad \text{and} \quad \mathbf{D} = \frac{1}{2}(\nabla \mathbf{v} + \nabla \mathbf{v}^T) \tag{3}$$

where ρ is the fluid density, $\mathbf{v}$ is the fluid velocity, $\mathbf{v}_m$ is the mesh velocity, $\mathbf{g}$ is the body force acting on the fluid, $\mathbf{T}$ is the total stress tensor in the fluid, $\mathbf{D}$ is the rate-of-strain tensor, p is the fluid pressure, and $\mathbf{I}$ is the identity tensor.

Capturing the motion and shape of the free surface is a difficult problem and there are a variety of approaches that can be used. In this work we adopt a boundary-fitted or "pseudo-solid" mesh mapping approach. In this technique the domain itself is modeled as a solid material capable of deforming under the action of various boundary constraints that are translated into suitable normal traction loads on the pseudo-solid. The displacements of this pseudo-solid material are obtained by solving the quasi-static equilibrium requirement for a solid material occupying the identical domain as the fluid,

$$\nabla \cdot \mathbf{T}_s + \mathbf{g}_s = 0\,, \tag{4}$$

where $\mathbf{T}_s$ is the total pseudo-solid stress and $\mathbf{g}_s$ is the body force per unit volume acting on the pseudo-solid. The previous equation (4) employs any

convenient constitutive equation for the pseudo-solid, for example,

$$\mathbf{T}_s = 2\mu_s \mathbf{E} + \lambda_s e \mathbf{I} \quad \text{and} \quad \mathbf{E} = \frac{1}{2}(\nabla \mathbf{d} + \nabla \mathbf{d}^T) , \tag{5}$$

where $\mathbf{E}$ is the small deformation strain tensor, $\mathbf{d}$ is pseudo-solid displacement, e is volumetric strain, and μ_s, λ_s are the Lame elastic coefficients. Other constitutive equations for the pseudo-solid can also be used, see [2].

Within the context of modelling the free-surface motion, application of this technique amounts to applying normal tractions on the fixed geometry boundaries of the domain so that they remain at their initial positions (although the mesh may slide tangentially along such boundaries) in conjunction with normal tractions on the pseudo-solid. The normal tractions on the pseudo-solid are derived from the essential physics at the as-yet undetermined interface location. In particular when the free surface has unit normal $\mathbf{n}_{fs}$ the kinematic constraint

$$\mathbf{n}_{fs} \cdot \mathbf{v} = 0 \tag{6}$$

establishes the interface as a material surface. Details of applying such *distinguishing conditions* are given in [3].

3 Numerical Method

Multiplying equations (1),(2) and (4) by weighting functions, integrating over the problem domain volume and applying conventional finite element method discretizations we obtain the Galerkin FEM equations. A number of different element interpolations are available in GOMA but for the problem studied here we employ eight-node hexahedral elements. As in many FEM applications the type of element used in the model is often dictated by a desire to control the total number of unknowns. For the class of problem studied here the numerical model yields seven unknown variables per node, three velocity components, three displacement components and the fluid pressure. In our model we employ trilinear interpolations for each of the field variables. This choice of elements violates the *Ladyshenka, Babuska & Brezzi* (LBB) stability condition [4] and additional measures must be taken to produce solutions. Here we use the Galerkin least-squares pressure stabilization technique of Droux & Hughes [5] to circumvent this difficulty. This technique involves the adding a weighted momentum equation residual to the weak form of the continuity equation thus introducing a stabilizing diagonal into the system matrix. While convergent for creeping flows [6], practical implementations of this stabilization technique neglect the small, higher-order, viscous stress terms with some slight mass balance error, depending upon the amount of stabilization employed, the Reynolds number, and whether inflow and outflow boundaries are present. Those small errors diminish with increased mesh refinement and are acceptable here until such

time as the preconditioners for iterative matrix solution of an LBB-based system are sufficient to the task.

Parallel solution of GOMA problems is effected by statically decomposing the finite element mesh using problem information to properly weight the equivalent corresponding unstructured graph. The problem graph is based largely on the mesh, but with vertex and edge weights apportioned according to the computational load and communications overhead incurred by placing the full multiphysics governing equations at each finite element node. The objective of the graph partition is to divide the graph into a specified number of pieces of equal computational work with a minimum of communications required when data must be exchanged between the processors that are responsible for each of the subdomains. Once the GOMA multiphysics problem description and the finite element mesh have been used to construct the equivalent graph, we use Chaco 2.0 [7] to partition the graph. While many options are available to select among the heuristic algorithms implemented within Chaco, the problems here have been decomposed using an initial, inertial partion followed by a local Kernighan-Lin refinement. For the problem sizes presented here, the partition took less than a minute and did not exceed 1 GB of physical memory. For much larger problems running on $O(10^3)$ processors, it is possible that the memory requirements to store and divide the original monolithic graph could exceed what is easily available on a single processor. Our decomposition utility has not yet been extended to run on a SPMD system which could handle the decomposition of very large problem graphs, although such extensions to graph partioners have been implemented [8].

Once the problem has been decomposed, each subdomain treats its problem as if it were global in all but a few instances. During the course of the Newton iteration and during the course of the iterative matrix solution the unknown vector must be updated to include the latest estimates from adjacent subdomains. Within GOMA, this communication is accomplished using the MPI message-passing library [9] and by categorizing degrees of freedom (DOF) on a given processor according to ownership of the associated finite element nodes as:

internal owned by this processor exclusively,

external owned by another processor, but needed on this processor,

boundary owned by this processor, but needed as an external node by one or more processors.

MPI user-derived datatypes are used to conveniently access the needed data structures, which remain static through the course of the calculation. User-defined datatypes are also used to facilitate the broadcast of a large repertoire of initialization information (over 550 distinct pieces, not counting array multiplicities.)

The solution of a linear matrix system in GOMA running in parallel entails the management of non-square systems that are both sparse, unsymmetric and unstructured. A generalization of the modified sparse row (MSR) format to distributed platforms is part of the Aztec matrix solver package [10] that GOMA employs. Aztec provides a convenient means for describing distributed matrix systems as well as a wealth of preconditioners and solvers that the user may activate depending on the problem. In addition, a block-based Variable Block Row (VBR) format is available within Aztec that permits more efficient solution of problems with many degrees of freedom per node [11].

By applying appropriate velocity, displacement, and traction boundary conditions that depend on the problem being solved, these discretized equations comprise a set of nonlinear algebraic residual equations. Within GOMA the governing residual equations are solved using successive Newton-Raphson iterations that update *all* of the unknowns simultaneously. A fully analytic representation of all of the Jacobian matrix entries provides for the most robust convergence of this scheme, despite the additional overhead of computing sensitivities with respect to mesh displacement unknowns. The Aztec linear solver package is used to solve the linear systems at each stage of the Newton iteration. For the example problem discussed here we employ a GMRES algorithm in conjunction with incomplete LU (ILU) preconditioning.

4 Test Problem

In this example problem we describe an application of GOMA's three-dimensional free-surface modeling capabilities to study fluid motion near the terminal edge portion of a slot coating device. Slot coating devices are widely used for applying thin films to long rolls of substrate material, *e.g.* photographic films or adhesive. The devices often consist of a long, narrow fluid feed slot with its exit a short distance away from the substrate to be coated. The substrate or web moves past the exit of the feed slot and perpendicular to the slot's long dimension. Fluid injected through the feed slot is entrained by the moving substrate, eventually becoming a thin coating on the substrate.

Because the length of the feed slot perpendicular to the direction of substrate motion is usually much greater than its width in the direction of substrate motion this process is often modeled as a two-dimensional problem. Most interest lies in the behavior of the bulk film layer overlying a large area of substrate, where the two-dimensional approximation is valid. Nonetheless, the coating flow in the vicinity of the feed slot edge can play an important role in determining the quality of the final film coating. Near this edge there exists a complicated three-dimensional free-surface fluid flow with both static and dynamic contact lines, where the fluid exits the slot

and where the fluid impinges the web, respectively. GOMA is unique in its ability to contend with problems of this nature and the slot coater edge flow provides a setting in which these capabilites can be demonstrated.

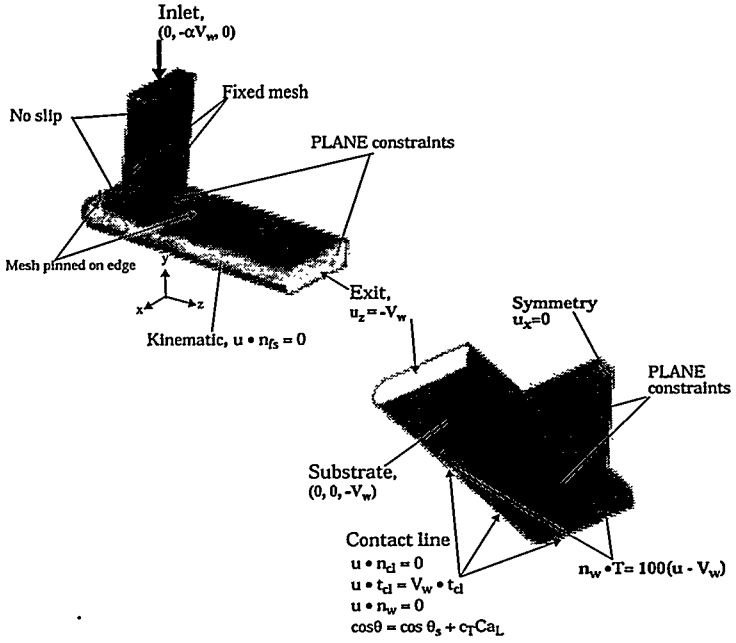

Figure 1: Slot coater edge flow geometry.

A representative edge geometry for the slot coater device is shown in Figure 1. This geometry represents only the region near the end of the slot coater and depicts the initial undeformed geometry prior to solving for the shape of the free surface. The remainder of the slot coater joins with the edge geometry in the plane labeled "Symmetry" where it is assumed that the normal variation of displacement and velocity along this length are zero. Coating fluid enters the domain at the "inlet" region, travels down the slot and exits onto a substrate that moves at velocity $\mathbf{V}_w$ in the negative z direction. We note that the end of the slot does not represent the beginning of the free surface but instead there is a flat region, perpendicular to the slot which separates its exit from the free surface. This flat region represents the portion of the confining, exterior walls of the slot itself that are in contact with the fluid. On the outer edges of this region the fluid is assumed to adhere on a static contact line that is fixed in space. The underside of the geometry represents a surface where the moving substrate contacts the coating flow. The substrate is not modeled explicitly but its presence is manifest via velocity boundary conditions imposed on the coating fluid.

The dynamic contact line is labeled in Figure 1 and is the curve where the fluid, the moving substrate, and the surrounding gas phase all meet.

Along a small narrow region of the flow adjacent to the contact line and surrounding the large area where the fluid contacts the moving substrate, the no-slip condition between web and fluid is not strongly enforced. Instead a Navier slip condition is used to allow the velocity field to transition from conditions imposed by the wetting line physics to the substrate speed. These wetting line velocity conditions have been discussed elsewhere [12]. Briefly, it is assumed that the wetting line physics impose a requirement that the velocity of fluid at the contact line be tangent to the contact line. Thus, if $\mathbf{n}_{\mathrm{cl}}$ is a unit vector normal to the contact line in the plane of the substrate and $\mathbf{t}_{\mathrm{cl}}$ is a unit vector tangent to the contact line in the same plane, then the two conditions imposed are

$$\mathbf{n}_{\mathrm{cl}} \cdot \mathbf{v} = 0 \qquad \text{and} \qquad \mathbf{t}_{\mathrm{cl}} \cdot \mathbf{v} = \mathbf{t}_{\mathrm{cl}} \cdot \mathbf{V}_{\mathrm{W}} \ . \tag{7}$$

An essential condition on the vertical velocity component (v_y) is used to impose impenetrability at the contact line. A fixed contact angle was applied at the contact line, where the angle is determined from the relationship between the web and free surface normals,

$$\mathbf{n}_{\mathrm{w}} \cdot \mathbf{n}_{\mathrm{fs}} = \cos\theta_{\mathrm{s}} \ . \tag{8}$$

The static contact angle, θ_{s}, was set to 11.47°, which was the initial contact angle occurring in the meshing solid model. A more complicated model for the dynamic contact angle which allows the contact angle to vary depending upon the local rate at which the contact line advances in its normal direction along the substrate is discussed in [12]. Over the remainder of the free surface the kinematic constraint,

$$\mathbf{n}_{\mathrm{fs}} \cdot \mathbf{v} = 0, \tag{9}$$

was imposed as noted above. Surface tension forces were also included in the fluid momentum equations applied to the free surface,

$$P_{\mathrm{ambient}} - \mathbf{n}_{\mathrm{fs}}\mathbf{n}_{\mathrm{fs}} : \mathbf{T}|_{\mathrm{fluid}} = 2\sigma\mathcal{H}, \tag{10}$$

where P_{ambient} is the ambient gas pressure, σ is the surface tension at the fluid/air interface, $2\mathcal{H}$ is the mean curvature of the surface. In this simulation the shear stress contributions in the gas phase are neglected, consistent with the low viscosity and density, while the ambient pressure P_{ambient} is taken as constant throughout the domain, though this latter condition is by no means a necessary one.

In general, except for the nodes on the free surface, all other nodes were fixed at their original positions. This was relaxed somewhat in the case of the substrate plane and the exit plane where the nodes were permitted to move within their initial plane but not normal to it. This allowed for appropriate deformation of the free surface and the dynamic contact line.

5 Analysis and Discussion

A finite element discretization model of the slot coater test problem containing roughly 50,000 unknowns was created using the SNL three-dimensional meshing utility, CUBIT [13]. This model was in turn used to generate the submodel decompositions required for parallel processing. The slot coater problem was analyzed using 1, 2, 4, 8, 16 and 32 processors on the ASCI Red Teraflops computer at SNL. Reconstructed results indicate that the solutions from each of the analysis were indistinguishable. For this fixed

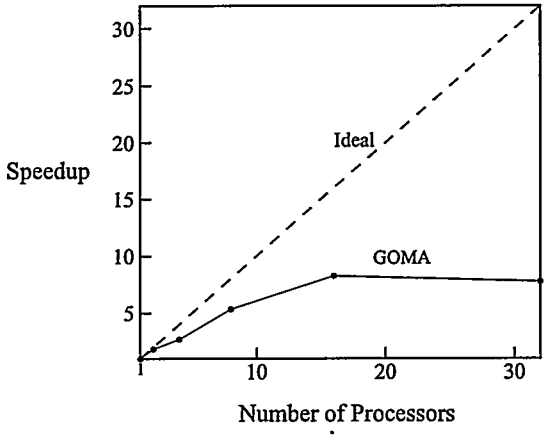

Figure 2: Speedup vs. number of processors for the slot coater test problem.

problem size, a plot of speedup versus number of processors for these analyses is shown in Figure 2. Neglecting the time taken to decompose and recompose the monolithic problem, the speedup on n processors, S_n, is defined simply as the ratio of the time taken to complete the simulation on a single processor, T_1, to the time taken to complete the simulation on many processors, T_n,

$$S_n = \frac{T_1}{T_n} \ . \tag{11}$$

The speedup, S_n, will typically increase with the number of processors, linearly in an ideal situation, depending on the algorithm and the communications overhead. A linear increase in S_n is equivalent to obtaining perfect parallel *efficiency*, E_n, defined as

$$E_n = \frac{S_n}{nS_1}, \tag{12}$$

which will be unity if the problem scales linearly.

From Figure 2 we find that less than linear speedup is obtained for the analyses performed. Reasonable speedup was obtained in going from one to up to 16 processors with the 32 processor result showing no speedup at all beyond 16 processors.

As in most parallel problems we expect that the best speedup is obtained when a large amount of the time on each processor is spent doing computation rather than communicating with other processors or spent idling due to an imbalanced workload. There are many ways to characterize the ratio of communication time to computation time, where the former may include message start-up costs as well as costs proportional to message length. As a rough estimate of relative communication to computational costs we consider the ratio, r_{cc}, of degrees of freedom communicated by a processor to the degrees of freedom owned by the processor

$$r_{cc} = \frac{N_{\text{external}} + N_{\text{boundary}}}{N_{\text{internal}} + N_{\text{boundary}}} \tag{13}$$

and plot that ratio versus the number of processors, Figure 3. Using the information from Figures 2 and 3 we can conclude that the parallel efficiency remains above 50% until $n \approx 16$ and $r_{cc} \approx 1$.

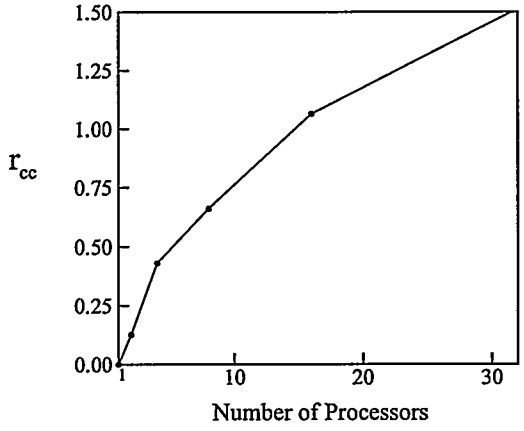

Figure 3: Ratio r_{cc} of DOF communicated by a processor to DOF owned by a processor for the slot coater test problem.

We note that while the same types of trends can be observed in other classes of problems it is difficult to arrive at universally optimal r_{cc} ratios.

	Number of Processors					
	1	2	4	8	16	32
Number of iterations	150	142	165	180	216	410

Table 1: Linear solve iterations for various number of processors.

In the context of the GOMA code this issue becomes clouded since the code deals with multiphysics problems and it is entirely possible that different processors will be solving different governing equations. The most important underlying factor that contributes to a loss of parallel efficiency as the number of processors increases is shown in Table 1. Here we find that the number of iterations required to solve the linear system at each Newton iteration increases with the number of processors. Basically, the more finely decomposed the global problem becomes, the slower the convergence becomes for a simple additive Schwarz scheme [14].

6 Summary

The parallel GOMA code has been used to analyze a free-surface incompressible flow problem. Based upon the preliminary scaling analyses performed here the usefulness of the parallel GOMA code has been demonstrated for problems run on up to about 8 processors, with the useful number of processors likely growing as the problem size increases beyond the 50,000 degrees of freedom in this case study. Significantly hampering the use of many processors for this class of problem is the concomitant growth in the number of iterations required to solve the linear system of equations at each Newton iteration. In future work we expect to reduce the iteration count by exploring optional overlapping preconditioners available in Aztec and, possibly, multilevel methods.

Practically speaking, the parallel version of GOMA will provide a tool for studying much larger problems than previously addressed. By decreasing the turnaround time for analysis, it should also prove useful for developing designs in which fully three-dimensional analysis of the flow is required.

Acknowledgements

Simulations were performed on the ASCI Red Intel Teraflops supercomputer at Sandia National Laboratories. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

References

[1] Schunk, P. R., Sackinger, P. A., Rao, R. R., Chen, K. S., Cairncross, R. A., Baer, T. A. & Labreche, D. A., GOMA 2.0 A Full-Newton Finite Element Program for Free and Moving Boundary Problems with Coupled Fluid/Solid Momentum, Energy, Mass, and Chemical Species Transport: User's Guide, SAND97–2404, *Sandia National Laboratories Technical Report* , Albuquerque, 1998.

[2] Cairncross, R. A., Schunk, P. R., Baer, T. A., Rao, R. R. & Sackinger, P. A., A finite element method for free-surface flows of incompressible fluids in three dimensions, part I: boundary-fitted mesh motion, to appear *Int. J. Numer. Meth. Fluids*, 2000.

[3] Sackinger, P. A., Schunk, P. R. & Rao, R. R., A Newton-Raphson pseudo-solid domain mapping technique for free and moving boundary problems: a finite element implementation, *J. Comp. Phys.*, **125**, 83–103, 1996.

[4] Oden, J. T. & Carey, G. F., *Finite Elements: Mathematical Aspects: Vol. IV*, Prentice-Hall, Englewood Cliffs NJ, 1983.

[5] Droux, J. J. & Hughes, T. J. R., A boundary integral modification of the Galerkin least squares formulation for the Stokes problem, *Comp. Methods Appl. Mech. Engrg.*, 113, 173-182, 1994.

[6] Hughes, T. J. R. & Franca, L. P., A new finite element formulation for computational fluid dynamics: VII. the Stokes problem with various well-posed boundary conditions: symmetric formulations that converge for all velocity/pressure spaces, *Comp. Meth. App. Mech. Eng.*, **65**, 85–96, 1987.

[7] Hendrickson, B. A. & Leland, R. W., The Chaco User's guide version 2.0, SAND95-2344, *Sandia National Laboratories Technical Report*, Albuquerque, 1995.

[8] Karypis, G. & Kumar, V., A parallel algorithm for multilevel graph partitioning and sparse matrix ordering, *Journal of Parallel and Distributed Computing*, **48** , 71–95, 1998.

[9] Snir, M., Otto, S., Huss-Lederman, S., Walker, D. & Dongarra, J. J., *MPI – The Complete Reference*, MIT Press, Cambridge Ma., 1998.

[10] Hutchinson, S. A., Prevost, L. V., Shadid, J. N. & Tuminaro, R. S., Aztec User's Guide Version 1.0, SAND95-1559, *Sandia National Laboratories Technical Report* , Albuquerque, 1995.

[11] Shadid, J., Hutchinson, S., Hennigan, G., Moffat, H., Devine, K. & Salinger, A. G., Efficient parallel computation of unstructured finite element reacting flow solutions, *Parallel Computing*, **23**, 1307–1325, 1997.

[12] Baer, T. A., Cairncross, R. A., Schunk, P. R., Rao, R. R. & Sackinger, P. A., A finite element method for free-surface flows of incompressible fluids in three dimensions, part II: dynamic wetting lines, to appear *Int. J. Numer. Meth. Fluids*, 2000.

[13] Blacker, T. D., Bohnhoff, W. J., Edwards, T. L., Hipp, J. R., Lober, R. R., Mitchell, S. A., Sjaardema, G. D., Tautges, T. J., Wilson, T. J., CUBIT Mesh Generation Environment–Volume 1: Users Manual, SAND94-1100, *Sandia National Laborotories Report*, Albuquerque, 1997.

[14] Smith, B., Bjorstad, P. & Gropp, W., *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations,* Cambridge University Press, Cambridge UK, 1996.