INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2009; **59**:801–815
Published online 13 June 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1852

Parallel three-dimensional simulation of the injection molding process

B. J. Araújo[1,*,†], J. C. F. Teixeira[1], A. M. Cunha[2] and C. P. T. Groth[3]

[1]*Departamento de Engenharia Mecânica, Universidade do Minho, Campus de Azurém, 4800-058 Guimarães, Portugal*
[2]*Departamento de Engenharia de Polímeros, Universidade do Minho, Campus de Azurém, 4800-058 Guimarães, Portugal*
[3]*University of Toronto Institute for Aerospace Studies, Toronto, Ont., Canada M3H 5T6*

SUMMARY

This paper describes the development of a parallel three-dimensional unstructured non-isothermal flow solver for the simulation of the injection molding process. The numerical model accounts for multiphase flow in which the melt and air regions are considered to be a continuous incompressible fluid with distinct physical properties. This aspect avoids the complex reconstruction of the interface. A collocated finite volume method is employed, which can switch between first- and second-order accuracy in both space and time. The pressure implicit with splitting of operators algorithm is used to compute the transient flow variables and couple velocity and pressure. The temperature equation is solved using a transport equation with convection and diffusion terms. An upwind differencing scheme is used for the discretization of the convection term to enforce a bounded solution. In order to capture the sharp interface, a bounded compressive high-resolution scheme is employed. Parallelization of the code is achieved using the PETSc framework and a single program multiple data message passing model. Predicted numerical solutions for several example problems are considered. The first case validates the solution algorithm for moderate Reynolds number flows using a structured mesh. The second case employs an unstructured hybrid mesh showing the capability of the solver to describe highly viscous flows closer to realistic injection molding conditions. The final case presents the non-isothermal filling of a thick cavity using three mesh sizes and up to 80 processors to assess parallel performance. The proposed algorithm is shown to have good accuracy and scalability. Copyright © 2008 John Wiley & Sons, Ltd.

Received 25 September 2007; Revised 23 April 2008; Accepted 24 April 2008

KEY WORDS: parallel computing; three-dimensional; finite volume method; mold filling; injection molding

*Correspondence to: B. J. Araújo, Departamento de Engenharia Mecânica, Universidade do Minho, Campus de Azurém, 4800-058 Guimarães, Portugal.
†E-mail: billy@dem.uminho.pt, billyaraujo@gmail.com

Contract/grant sponsor: Fundação para a Ciência e a Tecnologia (FCT)

1. INTRODUCTION

Injection molding is widely used for the production of plastic parts in many different manufacturing applications. Many of these parts have demanding technical specifications and are important components of products manufactured by a variety of industry sectors such as electronics, automobile, aerospace and telecommunications. Therefore, the optimization of both mold and plastic part design and the selection of appropriate processing conditions and parameters have become essential to obtain high-quality products without resorting to trial-and-error methods. Modifications incorporated in the design phase are systematically more economical and efficient.

The traditional approach to simulating the injection molding processes is to use a Hele-Shaw model in which the three-dimensional problem is represented by a set of mid-plane surfaces [1, 2]. These surfaces represent an imaginary shell located between the outer and inner surfaces of the part. This approach is suitable for the majority of plastic parts since thickness is usually small when compared with other dimensions. One advantage of this type of simulation is that the alteration of part thickness does not require the generation of a new mesh, since it is an attribute of each solution element and can be easily modified. This aspect can be particularly important in the case of optimizing part thickness to meet material and production costs. However, simulations based on the Hele-Shaw method also present several limitations, such as not being able to predict some flow-induced defects due to inertia. In addition, the model assumes a constant pressure throughout the thickness of the part and does not account for fountain flow effects. If one also considers the pre-processing requirements for obtaining a solution, mid-plane models are difficult to build and require significant user intervention to obtain high-fidelity meshes. This is particularly true for parts with many connecting ribs and large thickness variations. To overcome the limitations of Hele-Shaw models, fully three-dimensional simulations can be considered. However, one of the major disadvantages of this type of simulation is the often very large mesh size required to accurately represent the part. Unlike the Hele-Shaw model, where points across thickness can be added by simply increasing the number of layers, the three-dimensional model requires a significant increase in the size of the computational mesh. This leads to increases in both memory requirements and computational time to solve the problem, which can limit the application and prevent accurate three-dimensional simulations of complex cases. Parallel computing can be used to overcome these limitations by distributing the computational effort among multiple processors.

The development of numerical solution techniques for performing fully three-dimensional simulations of injection molding processes has occurred mainly along two slightly distinct paths: those solution techniques based on the finite element method (FEM) [3, 4] and others based on the finite volume method (FVM) [5]. The FEM provides good stability and accuracy for diffusion-dominant problems even using highly non-orthogonal meshes, while the FVM assures conservative fluxes, which is essential for convection-dominant problems. The first three-dimensional approaches to simulate the filling stage of the injection molding process used the FEM to discretize the governing flow equations. Inertial effects were neglected and a volume fraction equation was used to track the melt-front advancement. These approaches were later extended to simulate other stages of the injection molding process [6], as well as non-conventional injection molding processes such as gas-assisted injection molding and co-injection [7, 8]. Ilinca and Hétu [9] also developed a parallel FEM algorithm to simulate non-isothermal mold-filling. This approach included turbulence modeling, phase change effects and mold-cavity thermal resistance. Domain decomposition was achieved using the algorithms of Chaco and the METIS software package, and a parallel implementation was carried out for distributed memory architectures using MPI. The FVM can be

DOI: 10.1002/fld

applied to both structured and unstructured meshes. The type of mesh used will affect the scalability of the code. Structured meshes have better scaling properties although automatic generation of these meshes for complex geometries can be somewhat cumbersome. On the other hand, unstructured polyhedral meshes provide great geometric flexibility and the mesh generation process can be completely automated. Parallel computing along with FVM has been extensively used by the aerospace community to simulate internal and external high-speed flows using structured [10, 11] and unstructured meshes [12].

This paper considers the potential of applying a parallel unstructured FVM solution method to the simulation of the injection molding process. Details of the collocated FVM as well as the parallelization of the solution algorithm are described. Predictions of the proposed parallel algorithm are described for several example problems, demonstrating the capabilities of the method.

2. GOVERNING EQUATIONS

A mathematical model, consisting of a set of governing partial differential equations, is used herein to describe three-dimensional injection molding processes. Assuming that the polymer and the air behave as incompressible fluids during the filling process and that both can be approximated by a generalized Newtonian fluid, the melt and air flows can be described by the following partial differential equations for governing momentum and mass conservation:

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) - \nabla \cdot (\mu \nabla \mathbf{u}) = -\nabla p + \rho g \tag{1}$$

$$\nabla \cdot \mathbf{u} = 0 \tag{2}$$

where ρ is the density, $\mathbf{u}$ is the velocity vector, t is the time, μ is the dynamic viscosity, p is the pressure and g is the gravity vector. The temperature is taken to be governed by the scalar transport equation that includes both convection and conduction and is given by

$$\rho C_p \left(\frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right) - \nabla \cdot (k \nabla T) = S_T \tag{3}$$

where T is the temperature, C_p is the specific heat, k is the thermal conductivity and S_T is the heat source due to viscous heat dissipation:

$$S_T = \mu \dot{\gamma}^2 \tag{4}$$

where $\dot{\gamma}$ is the shear rate. The interface between the melt and air is captured using the volume-of-fluid (VOF) method. In this approach, a hyperbolic equation is used to represent the advection of a scalar, α, throughout the domain:

$$\frac{\partial \alpha}{\partial t} + \mathbf{u} \cdot \nabla \alpha = 0 \tag{5}$$

where the value of α indicates the fraction of fluid present in the cell. A value of zero indicates that the cell is totally occupied by air, while a value of one indicates that the cell is totally occupied

DOI: 10.1002/fld

by the polymer:

$$\alpha = \begin{cases} 0 & \text{air} \\ 1 & \text{polymer} \end{cases} \tag{6}$$

A value of α between zero and one indicates the presence of the interface, which is considered to be located at $\alpha = 0.5$. The physical properties in each cell are calculated by averaging the properties each fluid:

$$\psi = \alpha\psi_1 + (1-\alpha)\psi_0 \tag{7}$$

where ψ is the material property and the subscript indicates the type of fluid.

3. BOUNDARY AND INITIAL CONDITIONS

The domain is assigned the following initial conditions:

$$\mathbf{u} = 0, \quad p = 0, \quad T = T_{\text{wall}}, \quad \alpha = 0 \tag{8}$$

Since the system is incompressible, the polymer can only advance allowing the air to exit the domain. It is assumed that the air is able to exit the cavity walls. This is achieved applying a mixed wall/outlet boundary condition that acts as a wall for the polymer and as an outlet for the air inside the cavity. The air is not allowed to enter the cavity unless through an inlet boundary condition. This can be useful to simulate gas injection molding. The inlet has a velocity or pressure profile depending on the state of filling. During filling, the inlet may have an imposed velocity and then an imposed pressure to simulate the holding phase. The walls have an assigned temperature and thermal conductivity to account for the effect of using different molding materials. When enforcing boundary data in the finite volume discretization procedure described below, the terms associated with the known variables at the boundary faces are added to the right-hand side (RHS) of the equations.

4. NUMERICAL IMPLEMENTATION

An FVM is proposed and used to solve the preceding partial differential equations governing the flows of the polymer melt and air in injection molding processes. The method proposed here is suitable for use with unstructured meshes. Although these meshes are very flexible in terms of representing complex three-dimensional geometries, they are associated with the added inconvenience of an irregular data structure, which is also reflected in the structure of the matrices appearing in the solution procedure. The solution of sparse matrices with irregular structure usually requires more computational effort. However, these disadvantages are becoming less important as computational speed increases and sparse matrix solvers become more efficient.

In this work, the collocated FVM is employed to discretize the governing three-dimensional flow equations. This method has proved to be accurate and efficient when applied to the simulation of the injection molding process [5]. The domain is subdivided into a finite number of non-overlapping control volumes or cells of arbitrary topology.

DOI: 10.1002/fld

All of the governing equations for the melt and air can be re-cast into the form of the general transport equation given below with appropriate choice of the dependent variable ϕ, diffusivity Γ and source term S_ϕ:

$$\underbrace{\frac{\partial \rho\phi}{\partial t}}_{\text{unsteady term}} + \underbrace{\nabla\cdot(\rho\phi\mathbf{u})}_{\text{convective term}} - \underbrace{\nabla\cdot(\Gamma\nabla\phi)}_{\text{diffusive term}} = \underbrace{S_\phi}_{\text{source term}} \tag{9}$$

Integrating and applying the Gauss divergence theorem [13], Equation (9) becomes

$$\left(\frac{\partial \rho\phi}{\partial t}\right)_P + \frac{1}{V_P}\sum_{f=1}^{n} \rho_f \phi_f u_f A_f - \frac{1}{V_P}\sum_{f=1}^{n} \Gamma_f \nabla\phi_f A_f = S_\phi \tag{10}$$

where V_P is the volume of cell P, u_f is the face velocity and Γ_f is the face diffusivity. The values of the dependent variables at cell-face centers are calculated by linear interpolation, i.e. the central differencing scheme (CDS). This second-order scheme is more precise than the upwind differencing scheme (UDS) [14], but produces more oscillations becoming unstable for convective-dominant flows. Therefore, the UDS was used to discretize the convective term of the temperature equation to obtain a bounded solution and the CDS was used in other cases. Adding and rearranging all the terms present in Equation (10) and applying the implicit temporal discretization, the final set of equations can be expressed in the following form:

$$a_P\phi_P + \sum_{f=1}^{n} a_N\phi_N = S_\phi \tag{11}$$

In matrix form, Equation (11) can be re-expressed as

$$M\phi = b \tag{12}$$

where M is a sparse matrix, ϕ a vector of the dependent solution variables and b the RHS vector source term. This corresponds to the final algebraic equation, which links the value of the dependent variable at cell P with values at neighboring cells N. The governing equations are solved in a segregated manner in which the equations of momentum, continuity, temperature and volume fraction are solved sequentially requiring less computational memory. However, one of the issues when solving the equations using the segregated approach is to guarantee the coupling between velocity and pressure. To achieve this, the pressure implicit with splitting of operators algorithm is employed [15]. This method is an extension of the semi-implicit pressure-linked equations (SIMPLE) algorithm [16] and was developed to solve sequentially the discretized time-dependent flow equations. This method consists of a predictor and a corrector step. In the predictor step, a temporary velocity field is computed using the pressure field from the previous time step. In general, this temporary velocity field will not satisfy the zero-divergence condition. Therefore, the corrector step is employed to compute the correct pressure and velocity fields, which satisfy both continuity and momentum conservation. In terms of the time advancement of the solution, velocity and temperature equations are solved using the first-order explicit Euler time-marching scheme, while the volume fraction equation is solved using a second-order Crank–Nicholson method. The pressure equation is solved implicitly to ensure mass conservation. The compressive interface capturing scheme for arbitrary meshes (CICSAM) is employed to capture the interface between polymer and air [17]. This scheme uses a weighting factor to adjust between a complete upwind

DOI: 10.1002/fld

and downwind scheme. This weighting factor is calculated using a corrector–predictor method to maintain a sharp interface while ensuring a bounded volume fraction field. It is a high resolution scheme based on the normalized variable diagram, developed specifically for three-dimensional unstructured meshes of arbitrary topology.

5. PARALLEL COMPUTING

The proposed solution algorithm uses a single program multiple data message passing model, i.e. each process runs the same program and performs computations on its own subset of data. Dependent variables are stored in parallel vectors with ghost cell padding. One layer of ghost cells was used to share required data across parallel boundaries. Communication is performed when necessary to update information of the ghost cells. Matrices are stored in a compressed row format. In order to solve the system of equations, the flexible generalized minimum residual (GMRES) method [18] along with additive Schwarz preconditioning with one overlap was employed. Incomplete lower–upper with no additional fill (ILU(0)) factorization was used for local preconditioning of the linear equations.

Parallelization of the code was achieved within the PETSc framework [19]. PETSc is an open-source software package consisting of a suite of subroutines and data structures suitable for large-scale scientific applications requiring the solution of large, sparse, linear and nonlinear systems of equations on parallel and serial computers.

The allocation of solution data to each of the processors involved in the parallel computation is carried out by applying domain decomposition to the computational mesh. Domain decomposition is achieved using METIS [20], which performs the decomposition of unstructured meshes while minimizing the cutting surface and thus the amount of communication between processors [21]. The initial mesh is decomposed so as to assign one mesh partition per processor. The interface between connected partitions is assigned a boundary processor and contains a reference to the neighboring control volume and partition. This is useful when specifying information for parallel matrices. Each mesh is automatically renumbered conforming to PETSc requirements, i.e. all control volumes belonging to the same process are numbered sequentially. To reduce the bandwidth of the resulting matrices, the mesh can be first re-ordered using the reverse Cuthill–Mckee (RCM) algorithm [22]. This re-ordering strategy has been proved to increase the performance of the matrix solver by enhancing data locality [23, 24].

The parallel implementation of the injection molding simulation code was carried out on a Beowulf-class cluster with a total of 244 processors and 482 GB of distributed memory, using 1.5 GHz Itanium 2 processors with 6 MB of L3 cache.

6. NUMERICAL RESULTS

The open-source software package Gmsh [25] was used for pre- and post-processing of the solution data. This tool was used to generate the mesh, assemble the output from the various processors, combine time steps and process the results. Although the geometry is always three-dimensional, the numerical code developed in this work allows the simulation of one-, two- and three-dimensional flows. In one-dimensional cases, the mesh is divided only along one axis; in two-dimensional cases, the mesh is divided along two axes; and in the three-dimensional cases, the mesh is divided along

DOI: 10.1002/fld

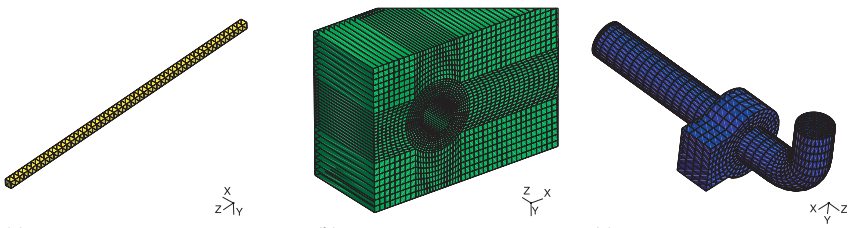

Figure 1. Mesh type: (a) one-dimensional; (b) two-dimensional; and (c) three-dimensional.

all three dimensions (see Figure 1). The mesh can be hybrid with cells of different topologies. The connectivity is established using an octree data structure to pair coincident faces.

6.1. Three-dimensional lid-driven cavity flow

Lid-driven cavity flow is one of the most studied fluid flow problems in computational fluid dynamics, as the geometry and boundary conditions are simple and can be easily reproduced. Despite these simplifications, lid-driven cavity flow contains some challenging flow physics making the problem appropriate for the validation of viscous flow solvers. The Reynolds number for the problem is defined as follows:

$$Re = \frac{\rho u_0 y_0}{\mu} \tag{13}$$

where y_0 is the height of the cubicle cavity and u_0 is the velocity of the lid. In this study flow was computed for three different Reynolds numbers with the cavity being divided into: (i) $65 \times 65 \times 65$ control volumes with $Re = 100$; (ii) $80 \times 80 \times 80$ control volumes with $Re = 400$; and (iii) $95 \times 95 \times 95$ control volumes with $Re = 1000$. The mesh was then decomposed and the simulation was performed in parallel using 48 processors. Since the flow is three-dimensional, the velocities in all three directions were calculated. The results obtained by the flow solver were compared with previous numerical results given by Wong and Baker [26]. As shown in Figure 2, there is good agreement between the current and previous results, particularly for low Reynolds numbers. For higher Reynolds number, the proposed methodology appears to slightly under-predict the flow recirculation in the cubicle cavity.

6.2. Three-dimensional mold filling of a thin part

This example shows the filling of a thin rectangular plate. The volume fraction equation was activated to estimate melt-front advancement. The hybrid unstructured mesh has three layers across thickness with a total of 900 hexahedra and 4806 prisms (see Figure 3). The viscosity of the polymer was considered to be constant ($\mu = 200\,\text{Pa s}$). Details of the molded part, material properties and processing conditions are presented in [5]. Figure 4 depicts the predicted pressure profile at the entrance of the cavity as a function of the filling time. It can observed that the predicted pressure profile is similar to the one presented by Chang and Yang [5]. In the narrow section of the part, the pressure increases rapidly and then increases more slowly as the filling changes from parallel to radial flow. After time $t = 0.56\,\text{s}$, filling is parallel again and pressure increase is approximately linear.

DOI: 10.1002/fld

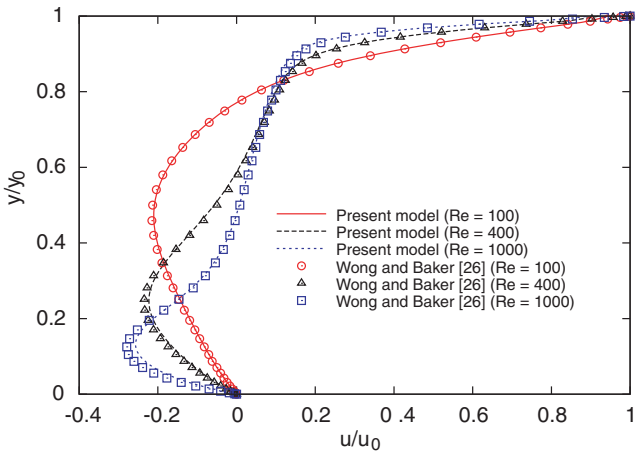

Figure 2. Velocity profiles at the centerline of the lid-driven cavity.

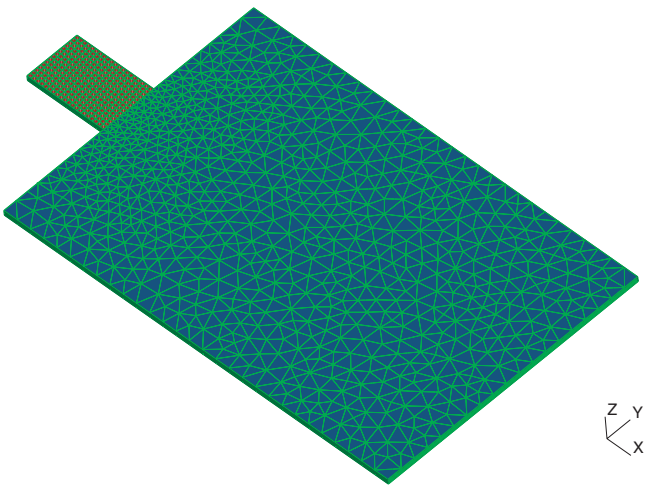

Figure 3. Unstructured hybrid mesh.

6.3. Three-dimensional mold filling of a thick part

In this last example, a polymer enters a thick rectangular cavity and has to flow around an obstacle. This illustrates the capability of the model to simulate flows with splitting and merging melt fronts. The other field variables such as temperature was also calculated using the transport equation described above.

The part has a length of 0.28 m, a width of 0.08 m and a thickness of 0.01 m. It also has a rectangular 0.04×0.08 m opening. The polymer flows through an entrance with an area of 0.01×0.005 m and a length of 0.005 m. The dimensions of the part are shown in Figure 5. A mesh

DOI: 10.1002/fld

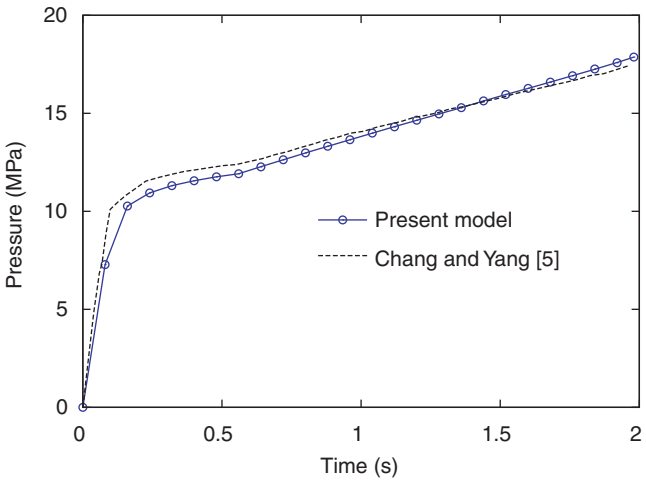

Figure 4. Comparison of the pressure profile at entrance.

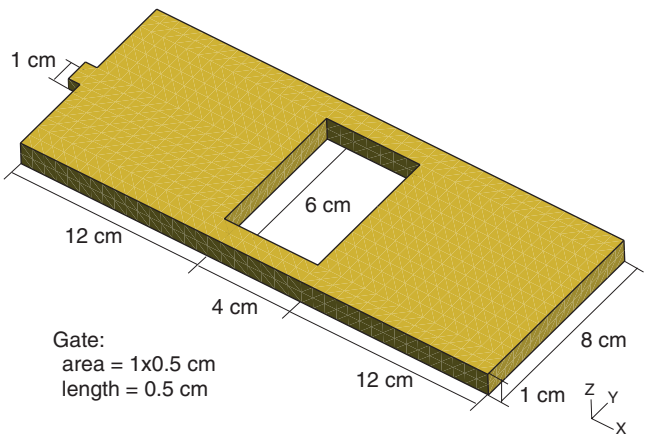

Figure 5. Geometry of the thick part.

Table I. Mesh sizes.

Case	Number of nodes	Number of cells
R2	11570	8656
R6	259038	233712
R8	598808	553984

parameter, r, was used to control the mesh refinement. Larger values of r lead to an increase in the mesh refinement and thus larger meshes. The following parameters were used: $r=2$ or (mesh R2), $r=6$ (mesh R6) and $r=8$ (mesh R8). The size of each mesh is presented in Table I.

DOI: 10.1002/fld

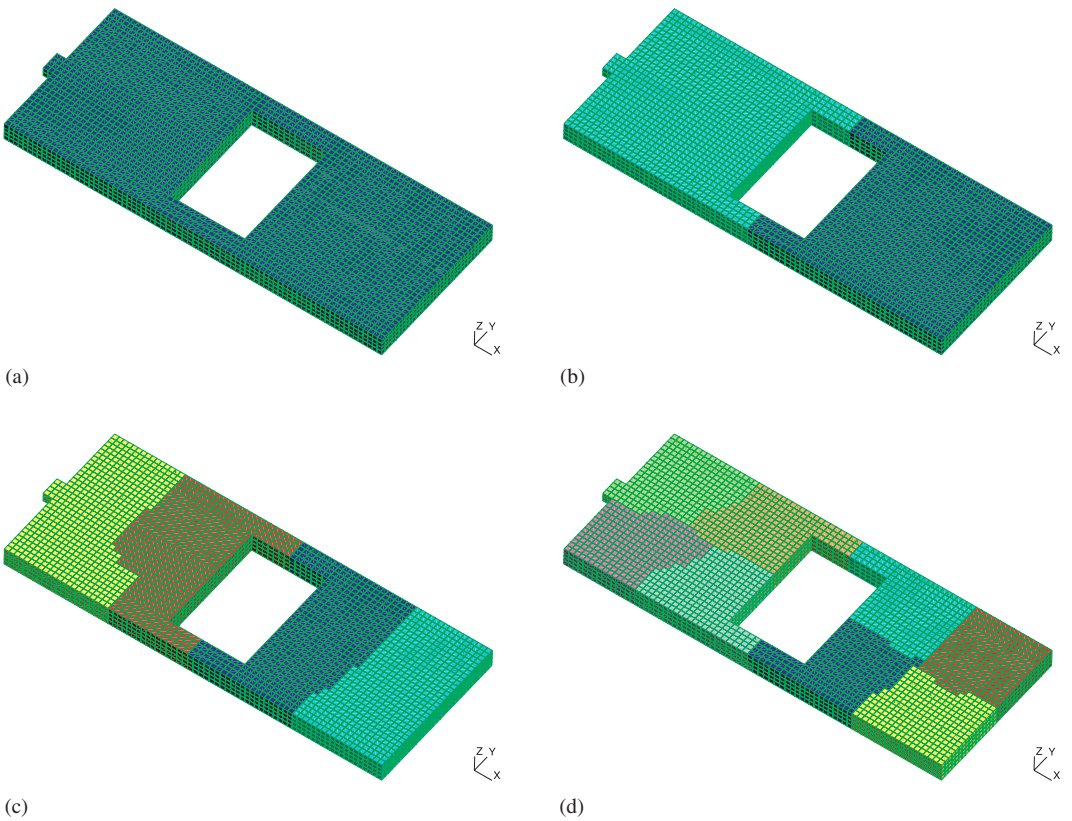

Figure 6. Mesh R2 and partitions: (a) 1 partition; (b) 2 partitions; (c) 4 partitions; and (d) 8 partitions.

In order to obtain an accurate temperature distribution across thickness, it is important to have a sufficient number of cells. Meshes R2, R6 and R8 have 4, 12 and 16 cells across thickness, respectively. Mesh R2 was decomposed into 2, 4 and 8 partitions (as shown in Figure 6), mesh R6 into 2, 4, 8, 16, 24 and 48 partitions and mesh R8 into 2, 4, 8, 16, 24, 48 and 80 partitions.

Mold filling simulations were refined both in space and in time. For case R2, the time interval was $\Delta t = 0.008\,\mathrm{s}$, for case R6, $\Delta t = 0.003\,\mathrm{s}$ and for case R8, $\Delta t = 0.002\,\mathrm{s}$. The time step is mostly limited by the VOF method, which requires a Courant number lower than 1. Higher Courant numbers will adversely affect the accuracy of the VOF method and thus the whole solution. The tolerance of the GMRES solver was set to 1×10^{-12} and the maximum number of iterations was set to 5000 for all cases. The effect of gravity was neglected in the calculations.

The polymer was considered to have a density of $\rho_1 = 1000\,\mathrm{kg/m^3}$ and a viscosity of $\mu_1 = 100\,\mathrm{Pa\,s}$. Since the properties of the air are not of great importance, it was considered as a pseudofluid with a density of $\rho_0 = 1 \times 10^{-3}\,\mathrm{kg/m^3}$ and a viscosity of $\mu_0 = 1\,\mathrm{Pa\,s}$ to improve convergence. The thermal conductivity of the polymer was considered as $k = 0.15\,\mathrm{W/(m\,^\circ C)}$ and the specific heat $C_p = 4186\,\mathrm{J/(kg\,^\circ C)}$. The melt was considered to enter the cavity with a constant velocity of $1\,\mathrm{m/s}$ and a temperature of 260°C. The surface of the mold was set at an uniform temperature of 30°C.

DOI: 10.1002/fld

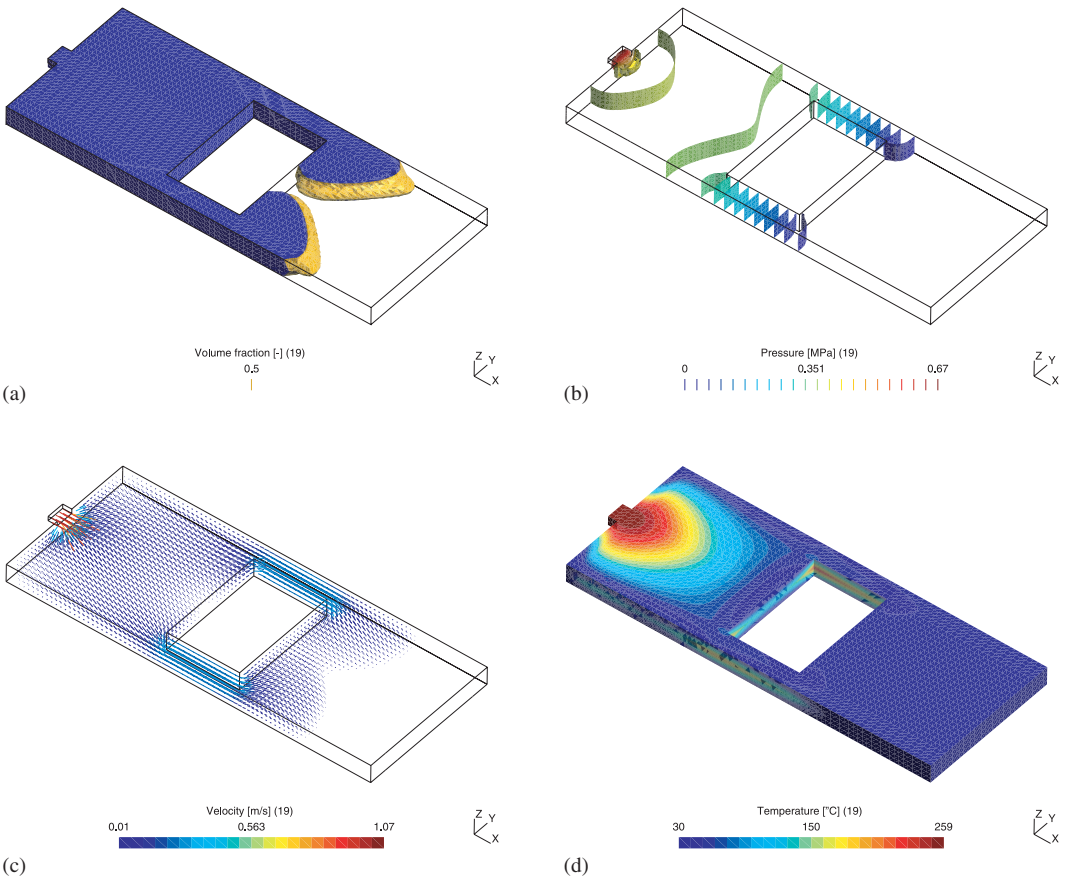

Figure 7. Results at $t = 1.9$ s—case R2: (a) volume fraction; (b) pressure field; (c) velocity field; and (d) temperature field.

In this case, the polymer viscosity was considered constant, unaffected by temperature and/or shear rate. Although this is a significant assumption, the purpose of this work was to assess the parallel efficiency of the code, not so much the effect of the temperature on the properties of the flow.

The complete filling of the part takes 2.73 s. This value corresponds to the theoretical filling time calculated using the part volume and flow rate. The predicted volume fraction, pressure, velocity and temperature fields at $t = 1.9$ s are presented in Figure 7. Figure 7(a) shows the three-dimensional surface ($\alpha = 0.5$), which represents the interface just before the union of both melt fronts. The union of the melt fronts is not affected by the air inside the cavity since it is allowed to escape through the mold walls. The pressure is presented in Figure 7(b) and is represented by isobaric surfaces. As expected, the velocity and pressure drops are higher around the inlet and the narrow sections of the part. Although the velocity is calculated in the entire solution domain, Figure 7(c) only shows the velocity vectors in the filled region. Figure 7(d) shows that the polymer is at a higher temperature close to the inlet and in the interior of the thick part.

DOI: 10.1002/fld

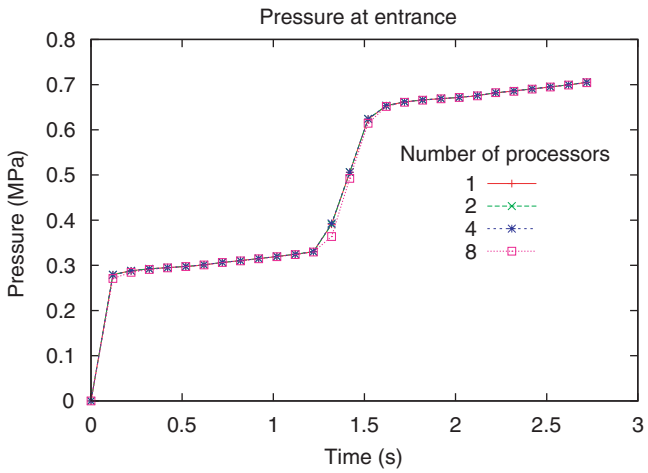

Figure 8. Comparison of the pressure profile at entrance.

The predicted low temperature near the interface may be related to the following factors:

1. the lower accuracy of the UDS used in the discretization of the convective term;
2. the high thermal conductivity that is applied to the faces at the boundary, which does not consider the much lower polymer thermal conductivity;
3. the temperature is imposed at the boundary;
4. the boundary temperature of the mold does not account for the cycle average temperature.

The accuracy of the temperature field could be improved using a bounded higher-order discretization scheme and/or using a different value for the thermal conductivity at the boundary taking also into account the thermal conductivity of the polymer. Instead of imposing a boundary temperature, a convective coefficient could be used or the boundary temperature could be initiated using a cycle average temperature considering cooling system design. An accurate solution of the three-dimensional temperature equation is important because it will affect other properties such as the density and viscosity of the polymer. It should be noted that these figures show the nodal values, although the solver determines the value of the dependent variables at cell center. The values at the boundary nodes are obtained by interpolation using the average cell center values. Therefore, these results on the surface of the part show the average value of control volume, which is at a certain distance from the boundary and not the boundary value itself.

The pressure at the entrance of the cavity is presented in Figure 8. It can be observed that the results are not affected by the parallelization strategy using 1, 2 and 4 processors. However, with 8 processors some small differences are noticeable in the pressure profile, more specifically when the flow enters the narrow sections of the part. This is mainly due to small variations in the VOF calculation using different number of processors which in turn is amplified by the pressure calculation. This aspect might be due to the higher Courant number at these sections, which produces error in the VOF method and will require further investigation.

The parallel speedup results for cases R2, R6 and R8 are shown in Tables II–IV. For case R2, the simulation time corresponds to the complete filling of the part (341 iterations), while for cases R6 and R8 it corresponds to 50 iterations. These same tests were run with a slower version of

DOI: 10.1002/fld

Table II. Parallel speedup—case R2.

Number of processors	Wall-clock time (s)	Speedup
1	381.11	1.00
2	202.12	1.89
4	103.21	3.69
8	59.93	6.36

Table III. Parallel speedup—case R6.

Number of processors	Wall-clock time (s)	Speedup
1	1181.59	1.00
2	690.97	1.71
4	400.33	2.95
8	199.57	5.92
16	101.63	11.63
24	74.26	15.91
48	43.45	27.19

Table IV. Parallel speedup—case R8.

Number of processors	Wall-clock time (s)	Speedup
1	2989.34	1.00
2	1762.71	1.70
4	976.30	3.06
8	497.77	6.01
16	269.42	11.10
24	184.63	16.19
48	104.12	28.71
80	70.34	42.50

the code and better scalability was achieved. By improving the speed of execution of the code, the parallel scalability decreased due to the higher proportion of the communication time relative to actual calculation time. The serial performance of the algorithm could be further improved implementing a more accurate volume fraction method, which does not depend so much on the Courant number as the CICSAM scheme and implementing a faster parallel solver for symmetric matrices such as a parallel conjugate gradient method or a parallel algebraic multigrid approach.

Although the speedup is reasonable using only 2 processors (1.89), the scalability degrades with the number of processors. Using 8 processors the speedup is 6.36. This corresponds to a parallel efficiency of 80%. By refining the mesh further (case R6), the scalability decreased (see Table III). In this case, the communication time represents a much higher fraction of the total simulation time. Using 8 processors, the speedup is 5.92 and this corresponds to a parallel efficiency of 74%. With 48 processors, the speedup is 27.19. For case R8, using 8 processors, the speedup is 6.01, which is similar to case R6. Using 48 and 80 processors, the speedup is 28.71 and 42.50, respectively.

DOI: 10.1002/fld

7. CONCLUSIONS

A new parallel algorithm has been developed for the non-isothermal three-dimensional mold filling. The proposed algorithm calculates the three-dimensional velocity, pressure and temperature field. It includes a VOF method to capture the position of the melt front. The solver was validated for three *Re* flow regimes 100, 400 and 1000 by considering three-dimensional lid-driven cavity flow. It was also applied to highly viscous flow problems representative of those associated with injection molding processes. In particular, the injection pressure was measured and compared during the filling of a thin rectangular cavity and parallel performance of the proposed algorithm was assessed for mold filling of a thick part. The results show that the algorithm has both good accuracy and reasonable parallel efficiency and scalability. Future research will involve further investigation and improvements of the thermal solver and the parallel efficiency of the algorithm to allow rapid solution of more complex industrial-scale simulations involving mesh consisting of millions of computational cells.

ACKNOWLEDGEMENTS

The author acknowledges the financial support from the Fundação para a Ciência e a Tecnologia (FCT) and the computer resources made available by the University of Toronto Institute for Aerospace Studies (UTIAS).

REFERENCES

1. Hieber CA, Shen SF. Flow analysis of the non-isothermal two-dimensional filling process in injection molding. *Israel Journal of Technology* 1978; **18**:577–582.
2. Kennedy P. *Flow Analysis of Injection Molds*. Hanser: Munich, 1995.
3. Talwar K, Costa F, Rajupalem V, Antanovski L, Friedl C. Three-dimensional simulation of plastic injection molding. *Proceedings of the Annual Technical Conference of the Society of Plastics Engineers ANTEC*, Atlanta, GA, 1998.
4. Hétu J-F, Gao DM, Garcia-Rejon A, Salloum G. 3D finite element method for the simulation of the filling stage in injection molding. *Polymer Engineering and Science* 1998; **38**:228–236.
5. Chang R-Y, Yang W-H. Numerical simulation of mold filling in injection molding using a three-dimensional finite volume approach. *International Journal for Numerical Methods in Fluids* 2001; **37**(2):125–148.
6. Ilinca F, Hétu J-F. Three-dimensional filling and post-filling simulation of polymer injection molding. *International Polymer Processing* 2001; **16**:291–301.
7. Ilinca F, Hétu J-F. Three-dimensional finite element solution of gas-assisted injection moulding. *International Journal for Numerical Methods in Engineering* 2002; **53**:2003–2017.
8. Ilinca F, Hétu J-F. Three-dimensional numerical modeling of co-injection molding. *International Polymer Processing* 2002; **17**:265–270.
9. Ilinca F, Hétu J-F. Simulation of 3-D mold-filling and solidification processes on distributed memory parallel architectures. *Proceedings of the ASME International Mechanical Engineering Congress and Exposition*. ASME: New York, 1997.
10. Sachdev JS, Groth CPT, Gottlieb JJ. Parallel AMR scheme for turbulent multi-phase rocket motor core flows. *Seventeenth AIAA Computational Fluid Dynamics Conference*. AIAA: New York, 2005; AIAA 2005-5334.
11. Northrup SA, Groth CPT. Parallel AMR scheme for turbulent multi-phase rocket motor core flows. *Forty-third AIAA Aerospace Sciences Meeting & Exhibit*. AIAA: New York, 2005; AIAA 2005-0547.
12. Mathur SR, Murthy JY. A pressure-based method for unstructured meshes. *Numerical Heat Transfer* 1997; **31**:195–215.
13. Ferziger JH, Peric M. *Computational Methods for Fluid Dynamics*. Springer: Berlin, 2002.
14. Spalding DB. A novel finite difference formulation for differential expressions involving both first and second derivatives. *International Journal for Numerical Methods in Engineering* 1972; **4**(4):551–559.

DOI: 10.1002/fld

15. Issa RI. Solution of the implicitly discretised fluid flow equations by operator-splitting. *Journal of Computational Physics* 1985; **68**(1):40–65.
16. Patankar SV. *Numerical Heat Transfer and Fluid Flow*. McGraw-Hill: New York, 1980.
17. Ubbink O, Issa RI. A method for capturing sharp fluid interfaces on arbitrary meshes. *Journal of Computational Physics* 1999; **153**(1):26–50.
18. Saad Y, Schultz MH. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing* 1986; **7**(3):856–869.
19. *PETSc—Portable*, *Extensible Toolkit for Scientific Computation*. http://www-unix.mcs.anl.gov/petsc/petsc-as/ [01 September 2006].
20. *METIS—Serial Graph Partitioning and Fill-reducing Matrix Ordering*. http://glaros.dtc.umn.edu/gkhome/metis/metis/overview [15 August 2003].
21. Karypis G, Kumar V. Multilevel *k*-way partitioning scheme for irregular graphs. *Journal of Parallel and Distributed Computing* 1998; **48**(1):96–129.
22. *Reverse Cuthill Mckee* (RCM). http://www.math.temple.edu/∼daffi/software/rcm/ [24 September 2005].
23. Blanco M, Zingg DW. A fast solver for the Euler equations on unstructured grids using a Newton-GMRES method. *Thirty-fifth AIAA Aerospace Sciences Meeting & Exhibit*. AIAA: New York, 1997; AIAA 1997-331.
24. Gropp WD, Kaushik DK, Keyes DE, Smith BF. High-performance parallel implicit CFD. *Parallel Computing* 2001; **27**:337–362.
25. *Gmsh*: *A Three-dimensional Finite Element Mesh Generator with Built-in Pre- and Post-processing Facilities*. http://geuz.org/gmsh/ [14 May 2006].
26. Wong KL, Baker AJ. A 3D incompressible Navier–Stokes velocity–vorticity weak form finite element algorithm. *International Journal for Numerical Methods in Fluids* 2002; **38**:99–123.

DOI: 10.1002/fld