Parallel Computing Investigations for the Projection Method Applied to the Interface Transport Scheme of a Two-phase Flow by the Method of Characteristics

Mireille Haddad † ‡, Toni Sayah, Pierre-Henri Tournier, Frédéric Hecht

PARALLEL COMPUTING INVESTIGATIONS FOR THE PROJECTION METHOD APPLIED TO THE INTERFACE TRANSPORT SCHEME OF A TWO-PHASE FLOW BY THE METHOD OF CHARACTERISTICS.

MIREILLE HADDAD[†‡], FRÉDÉRIC HECHT[†], TONI SAYAH[‡], AND PIERRE HENRI TOURNIER[†]

ABSTRACT. This paper deals with the discretization of the problem of mould filling in iron foundry and its numerical solution using a Schwarz domain decomposition method. An adapted technique for domain decomposition methods that suits the discretization in time by the method of characteristics is introduced. Furthermore, the projection method is used to reduce the computation time. Finally, numerical experiments show and validate the effectiveness of the proposed scheme.

KEYWORDS. Two-phase flow, Level-Set function, finite element method, method of characteristics, domain decomposition methods, overlapping decomposition, parallel computing.

1. INTRODUCTION.

Serious limitations in terms of computational time and memory requirement are encountered in modeling the problem of mould filling with medium and large series shapes by the finite element method. The algorithm built for the simulation of the problem in ([?]) was designed with the priority of reducing the computational cost. The problem was discretized in time by the first order method of characteristics combined with a time splitting approach where the discretization in space was used with P1 finite elements. However, this algorithm still remains inefficient for large domains.

In the past, improving performance of such simulations, either in speed or in the amount of data processed, was only a matter of waiting for the next generation of processors. But since approximately year 2005 the clock speed stagnates at 2-3 GHz. The increase in performance is almost entirely due to the increase in the number of cores per processor. To improve the performance of an algorithm, parallelism must be used.

There exists in the litterature direct and iterative solvers to parallelize large problems. Contrary to direct methods, the appealing feature of domain decomposition methods is that they are naturally parallel. Also, these iterative solvers are highly memory efficient but they are highly problem specific. For example, they do not perform well for problems involving highly ill conditioned matrices. On the other hand, direct solvers are more robust. However, large memory requirements pose a serious constraint for the direct methods. Many ways to circumvent this problem are being investigated. The development of multifrontal solvers ([?]) has enormously improved the computational efficiency and lowered the memory requirements for direct solvers [?], [?].

Recently a parallel direct solver, Multifrontal Massively Parallel Solver (MUMPS) (See [?], [?] and [?]) in a distributed environment [?] has been studied. It has been reported that by using MUMPS solver, larger problems can be solved in a parallel environment as the memory is distributed amongst the different processors. However it has also been reported that the scalability of MUMPS solver [?] is not very high. In order to obtain good scalability, instead of using a parallel sparse direct solver, the problem itself is split amongst the different processors using domain decomposition [?] and a highly efficient sparse direct solver is used in each of subdomain problems. Schwarz methods are the oldest domain decomposition methods. They were invented by Hermann Amandus Schwarz in 1869 as an analytical

tool to rigorously prove results obtained by Riemann through a minimization principle (see [**?**]). Renewed interest in these methods was sparked by the arrival of parallel computers, and variants of the method have been introduced and analyzed (for a historical presentation of these kind of methods see [**?**]).

In this paper, we will use the domain decomposition methods that are one of the dominant paradigms in contemporary large scale partial differential equation simulations to parallelize the problem of mould filling in iron foundry in the context of overlapping methods. The problem is discretized by the finite element method in space and by the method of characteristics in time. This discretization in time allows us to relax the CFL condition and to use a relatively large time step in comparison with other methods, resulting in a significant decrease of the number of iterations to simulate the filling of the moulds, cutting down simulation time tremendously. However, applying this method next to the boundaries of each subdomain in the domain decomposition method creates small numerical errors that accumulate and add unwanted numerical diffusion especially with a time step greater than the space mesh step.
In this paper, we introduce a technique to reduce these numerical errors. The performance of the algorithm is assessed by numerical simulations executed on hundreds of cores, for solving various highly heterogeneous elliptic problems in 3D with millions of degrees of freedom.

The outline of the paper is as follows:

- In section 2, we recall the problem of mould filling.
- In section 3, we present the Schwarz algorithm associated with the method of characteristics.
- In section 4, we show numerical results for validation.

2. Problem of mould filling

In this section, we consider the problem of mould filling for an iron foundry. The objective is to model and solve the corresponding industrial problem which requires a huge computational cost in memory and time. In this work we will follow the mathematical model introduced and treated in [**?**].

We consider an unsteady and laminar flow of two immiscible, viscous, Newtonian and of large density ratio fluids. In addition, the fluids are considered incompressible and isothermal, thus neglecting the variations of density and viscosity due to changes in pressure and temperature. Furthermore, by assuming that both fluids are homogenous, we suppose that the viscosity and density are constant in each fluid. We assume the interface is sharp and impermeable, thus the mass transfer across the interface is neglected (See [**?**] and [**?**]).

2.1. **System of equations.** We consider an interval $[0, T] \subset \mathbb{R}$, where T is a positive real number, and an arbitrary time $t \in [0, T]$. Let Ω be a bounded simply connected open domain in $\mathbb{R}^d$, $d = 2, 3$, with a Lipschitz-continuous connected boundary $\partial\Omega$. We denote by $\mathbf{n}$ the outward unit normal vector to the interface $\partial\Omega$ and $(\mathbf{e_1}, \mathbf{e_2})$ the canonical base of $\mathbb{R}^2$ (respectively $(\mathbf{e_1}, \mathbf{e_2}, \mathbf{e_3})$ the canonical base of $\mathbb{R}^3$).

We suppose that at each time $t \in [0, T]$, Ω is divided into two open sub-domains Ω_1 and Ω_2 evolving in time and separated by the interface Γ such that $\overline{\Omega} = \overline{\Omega}_1(t) \cup \overline{\Omega}_2(t)$ and $\Omega_1(t) \cap \Omega_2(t) = \emptyset$.

We denote by $\partial\Omega_i$ the boundary Ω_i, $i = 1, 2$, which is divided into four parts such that $\partial\Omega_1 \cap \partial\Omega_2 = \Gamma(t)$, $\partial\Omega = \Gamma_0 \cup \Gamma_1 \cup \Gamma_2$ where Γ_0 is the bottom of the boundary as indicated in Figure **??** (corresponding to the inlet), Γ_2 is the top of the boundary (corresponding to the free boundary of the fluid) and $\Gamma_1 = \partial\Omega \setminus (\Gamma(t) \cup \Gamma_0 \cup \Gamma_2)$ (corresponding to the wall).
The interface is captured, at each time step, by the advection of the Level-Set function [**?**], denoted by ϕ, by the fluid velocity denoted by $\mathbf{u}$. The level set function is a signed distance function which is positive
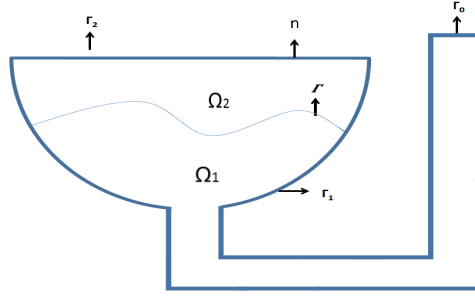
FIGURE 1. An arbitrary domain Ω

in Ω_2 and negative in Ω_1. In [?], we describe the mathematical model of equations corresponding to ϕ:

$$\partial_t \phi + \mathbf{u}.\nabla\phi - \varepsilon\Delta\phi = 0 \qquad \text{in } \Omega, \, , \tag{2.1a}$$

$$\partial_\mathbf{n}\phi = G(\mathbf{u}) \qquad \text{on } \Gamma_1, \tag{2.1b}$$

$$\partial_\mathbf{n}\phi = 0 \qquad \text{on } \Gamma_0 \cup \Gamma_2, \tag{2.1c}$$

$$\phi(\mathbf{x},0) = \phi_0 \qquad \text{in } \Omega, \tag{2.1d}$$

$$\tag{2.1e}$$

where ϕ_0 is the initial position of the interface and $G(\mathbf{u}) = \dfrac{\Big(\rho g\mathbf{e_d} + \rho\big((\mathbf{u}.\nabla)\mathbf{u}\big)\Big).\mathbf{n}}{\Big\|\big(\rho g\mathbf{e_d} + \rho\big((\mathbf{u}.\nabla)\mathbf{u}\big)\big).\mathbf{n}\Big\|}$ is a boundary

condition under gravitational effect. The term $\varepsilon\Delta\phi$ is the artificial viscosity that vanishes as $\varepsilon \to 0$. For the discussion of this introduced term and specially in the context of numerical investigations corresponding to this work, we refer to our previous work [?].
The density ρ and the viscosity μ can then be written in Ω as:

$$\rho(\mathbf{x},t) = \rho_1(\phi \leq 0) + \rho_2(\phi > 0) \text{ and } \mu(\mathbf{x},t) = \mu_1(\phi \leq 0) + \mu_2(\phi > 0)$$

where $\rho_i, i = 1, 2$ (respectively μ_i) is the density (respectively the viscosity) in $\Omega_i, i = 1, 2$.

The fluids motion is described by the following Navier-Stokes problem [?]: at each time $t \in]0, T[$,

$$\rho\Big(\frac{\partial\mathbf{u}}{\partial t} + (\mathbf{u}.\nabla)\mathbf{u}\Big) - \text{div}(2\mu\mathbb{D}\mathbf{u}) + \nabla p = \mathbf{f} \qquad \text{in } \Omega_1 \cup \Omega_2 \,, \tag{2.2a}$$

$$\text{div }\mathbf{u} = 0 \qquad \text{in } \Omega, \tag{2.2b}$$

$$\mathbf{u} = U_{in} \qquad \text{on } \Gamma_0, \tag{2.2c}$$

$$\beta^{-1}\mathbf{u}.\mathbf{n} + {}^t\mathbf{n}(2\mu\mathbb{D}\mathbf{u} - pI).\mathbf{n} = 0 \qquad \text{on } \Gamma_1 \tag{2.2d}$$

$$\alpha\mathbf{u}.\boldsymbol{\tau} + {}^t\mathbf{n}(2\mu\mathbb{D}\mathbf{u} - pI).\boldsymbol{\tau} = 0 \qquad \text{on } \Gamma_1 \tag{2.2e}$$

$$(2\mu\mathbb{D}\mathbf{u} - pI).\mathbf{n} = 0 \qquad \text{on } \Gamma_2, \tag{2.2f}$$

$$[2\mu\mathbb{D}\mathbf{u} - pI].\mathbf{n} = 0 \qquad \text{on } \Gamma, \tag{2.2g}$$

$$[\mathbf{u}] = 0 \qquad \text{on } \Gamma, \tag{2.2h}$$

$$\mathbf{u}(\mathbf{x},0) = 0 \qquad \text{in } \Omega, \tag{2.2i}$$

where $\mathbf{f} = -\rho g\mathbf{e_d}$ is the gravitational force vector, I is the identity matrix, β is a penalty coefficient, which is a small number [?], and U_{in} designates the velocity of the flux at the inlet. The equation (2.2a) is the Navier-Stokes equation which describes the fluids motion in the domains Ω_1 and Ω_2. The equation (2.2b) is the incompressibility condition. The equation (2.2c) is the non-homogeneous Dirichlet boundary condition at the inlet Γ_0. The equations (2.2d) and (2.2e) are the Navier boundary condition [?] at the wall Γ_1 where α is the friction coefficient and β is a penalty coefficient, which is a small number [?]. The equation (2.2f) is the do-nothing boundary condition at the outlet Γ_2. The equations

(2.2g) and (2.2h) are the jump condition [?] on the interface Γ. The equation (2.2i) is the initial condition.

2.2. Weak variational formulation.
In order to write the variational formulation of the previous problem, we introduced the following Sobolev spaces (m and $p \in \mathbb{N}$):

$$W^{m,p}(\Omega) = \{\mathbf{v} \in L^p(\Omega), \partial^\alpha \mathbf{v} \in L^p(\Omega), \forall |\alpha| \leq m\},$$

$$H^m(\Omega) = W^{m,2}(\Omega),$$

equipped with the following semi-norm and norm:

$$|\mathbf{v}|_{m,p,\Omega} = \{\sum_{|\alpha|=m} \int_\Omega |\partial^\alpha \mathbf{v}(x)|^p dx\}^{\frac{1}{p}}$$

and

$$\|\mathbf{v}\|_{m,p,\Omega} = \{\sum_{K \leq m} |\mathbf{v}|^p_{k,p,\Omega}\}^{\frac{1}{p}}.$$

We considered the following spaces:

$$X_u = \{\mathbf{u} \in H^1(\Omega) \ / \ \mathbf{u} = U_{in} \ \ \text{on } \Gamma_0\},$$
$$X_v = \{\mathbf{v} \in H^1(\Omega) \ / \ \mathbf{v} = 0 \ \ \text{on } \Gamma_0\},$$
$$M = L_0^2(\Omega) = \{p \in L^2(\Omega) \ / \ \int_\Omega p(x)dx = 0\},$$
$$Y = H^1(\Omega).$$

and we supposed that $\mathbf{f} \in L^2(\Omega)^d$.

Then, the variational formulation can be written as [?]:
Find $(\mathbf{u}, p) \in X_u \times M$, $\phi \in Y$ such that for all $(\mathbf{v}, q) \in X_v \times M$ and $\forall r \in Y$

$$\begin{cases} \int_\Omega \left(\frac{\partial \phi}{\partial t} + \mathbf{u}.\nabla\phi\right)r + \varepsilon \int_\Omega \nabla\phi\nabla r - \varepsilon \int_{\Gamma_1} G(\mathbf{u})r = 0. \\ \rho \int_\Omega \left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}.\nabla)\mathbf{u}\right)\mathbf{v} + \mu \int_\Omega \nabla\mathbf{u}\nabla\mathbf{v} - \int_\Omega p\,\text{div}\,\mathbf{v} \\ \qquad\qquad - \int_{\Gamma_1} \beta^{-1}\, {}^t\mathbf{u}(\mathbf{n}\ {}^t\mathbf{n})\mathbf{v}ds - \int_{\Gamma_1} \alpha\, {}^t\mathbf{u}(\boldsymbol{\tau}\ {}^t\boldsymbol{\tau})\mathbf{v}ds = -\int_\Omega \rho g\mathbf{e_d}\mathbf{v}. \\ \int_\Omega \text{div}\,\mathbf{u}\ q = 0. \end{cases} \qquad (2.3)$$

3. Discrete system

In this section, in order to simulate the model, we discretize the problem in time by the method of characteristics [?] and in space by the finite element method.

3.1. Time and space discretizations.
We use a time splitting approach and we discretize the problem (??) in time by using the method of characteristics. Thanks to this method, it is theoretically possible to follow the particles over time along their trajectory by solving, for each particle, an ordinary differential equation called characteristic equation:

$$\begin{cases} \frac{\partial X}{\partial t}(\mathbf{x}, s; t) = \mathbf{u}(X(\mathbf{x}, s; t), t), \\ X(\mathbf{x}, s; s) = x, \end{cases} \qquad (3.1)$$

where the characteristic curve $X(\mathbf{x}, s; t)$ denotes the position at time t of a fluid particle located at position $\mathbf{x}$ at time s.

We introduce a partition of the interval $[0, T]$ into N subintervals $[t_n, t_{n+1}]$, such that $\Delta t = \dfrac{T}{N}$ and the points $t_n = n\Delta t$, for $n = 0, ..., N$.

Using the following approximation of the total derivative along the characteristic curves, we approximate $\dfrac{D\mathbf{u}}{Dt}$ and $\dfrac{D\phi}{Dt}$ at time $t = t^{n+1}$ by:

$$\frac{D\mathbf{u}}{Dt}(x, t^{n+1}) \approx \frac{\mathbf{u}(x, t^{n+1}) - \mathbf{u}(X^n(x), t^n)}{\Delta t} \tag{3.2}$$

and

$$\frac{D\phi}{Dt}(x, t^{n+1}) \approx \frac{\phi(x, t^{n+1}) - \phi(X^n(x), t^n)}{\Delta t} \tag{3.3}$$

where $X^n(x)$ is the approximation of $X(x, t^{n+1} : t^n)$.

For the discretization in space, we consider τ_h a regular family of triangulations of Ω (by triangles in 2D and tetrahedra in 3D) of parameter h.

We introduce the discrete spaces $X_{u,h} \subset X_u$, $X_{v,h} \subset X_v$, $M_h \subset M$ and $Y_h \subset Y$ such that:

$$X_{uh} = \{\mathbf{u}_h \in X_{\mathbf{u}}; \forall k \in \tau_h; \mathbf{u}_{h|k} \in P_b(k)^d\},$$
$$X_{vh} = \{\mathbf{v}_h \in X_{\mathbf{v}}; \forall k \in \tau_h; \mathbf{v}_{h|k} \in P_b(k)^d\},$$
$$M_h = \{q_h \in M \cap C^0(\Omega); \forall k \in \tau_h, q_{h|k} \in P_1(k)\},$$
$$L_h = \{\mathbf{u}_h \in L^2(\Omega); \forall k \in \tau_h; \mathbf{u}_{h|k} \in P_1(k)^d\},$$

where $P_1(k)$ is the Lagrange finite element of degree 1 on the element k and $P_b(k)$ is spanned by functions in $P_1(k)$ and the bubble function on k (for each element k, the bubble function is equal to the product of the barycentric coordinates associated with the vertices of k).

We denote by $\mathbf{u}_h^{n+1}$, p_h^{n+1} and ϕ_h^{n+1} respectively the discrete velocity, pressure and Level-Set function at time t_{n+1}. The discrete system corresponding to the variational formulation can be written in the following form:

Having $\mathbf{u}^n \in X_{u,h}$ and $\phi_h^n \in Y_h$, find $(\mathbf{u}_h^{n+1}, p_h^{n+1}) \in X_{u,h} \times M_h$ and $\phi_h^{n+1} \in Y_h$ such that

$$
\begin{cases}
\left(\dfrac{\phi_h^{n+1} - \phi_h^n o X^n}{\Delta t}, r_h\right) + (\varepsilon \nabla \phi_h^{n+1}, \nabla r_h) - \varepsilon \displaystyle\int_{\Gamma_1} G(\mathbf{u}_h^n) r_h = 0 & \forall r_h \in Y_h, \\[2ex]
\rho_h^n \left(\dfrac{\mathbf{u}_h^{n+1} - \mathbf{u}_h^n o X^n}{\Delta t}, \mathbf{v}_h\right) + \mu_h^n \, (\nabla \mathbf{u}_h^{n+1}, \nabla \mathbf{v}_h) - (p_h^{n+1}, \operatorname{div} \mathbf{v}_h) & \\[1ex]
\quad - \displaystyle\int_{\Gamma_1} \alpha \, {}^t\mathbf{u}_h^{n+1}(\boldsymbol{\tau} \; {}^t\boldsymbol{\tau}) \mathbf{v}_h ds - \int_{\Gamma_1} \beta^{-1} \, {}^t\mathbf{u}_h^{n+1}(\mathbf{n} \; {}^t\mathbf{n}) \mathbf{v}_h ds = (-\rho_h^n g \mathbf{e_d}, \mathbf{v}_h) & \forall \mathbf{v} \in X_v, \\[2ex]
(q_h, \operatorname{div} \mathbf{u_h^{n+1}}) = 0 & \forall q_h \in M_h.
\end{cases}
\tag{3.4}
$$

3.2. Algorithm of the projection method.

Our purpose is to model mould filling for an iron foundry. The mould that we will use are of medium and large series. The computational cost in time and memory is rather very high. In order to reduce it, we use a time splitting approach which is also called the projection method (See [?], [?]). The algorithm corresponding to this method was introduced in [?]. It corresponds to computing first an intermediate velocity $\mathbf{u}^*$ that does not satisfy the incompressibility condition, then projecting this intermediate velocity on the set of divergence free functions to get the value of the pressure solution of the problem, and finally computing the correct velocity from the obtained results. The discrete variational formulation can be written in the following form: Having $\mathbf{u}^n \in X_{u,h}$ and $\phi_h^n \in Y_h$, find $(\mathbf{u}_h^{n+1}, p_h^{n+1}, \phi_n^{n+1}) \in X_{u,h} \times M_h \times Y_h$ following the steps:

1- Compute the intermediate velocity $\mathbf{u}^*$: Find $\mathbf{u}_h^* \in X_{uh}$ such that for all $\mathbf{v}_h \in X_{vh}$, we have:

$$
\int_\Omega \rho_h^n \frac{\mathbf{u}_h^* - \mathbf{u}_h^n o X^n}{\Delta t} \mathbf{v}_h + \int_\Omega \mu_h^n \nabla \mathbf{u}_h^* \nabla \mathbf{v}_h + \int_\Omega \nabla p^n \mathbf{v}_h - \int_{\Gamma_1} \alpha \, {}^t\mathbf{u}_h^{n+1}(\boldsymbol{\tau} \; {}^t\boldsymbol{\tau}) \mathbf{v}_h ds
$$
$$
- \int_{\Gamma_1} \beta^{-1} \, {}^t\mathbf{u}_h^{n+1}(\mathbf{n} \; {}^t\mathbf{n}) \mathbf{v}_h ds = \int_\Omega \rho_h^n g \mathbf{e_d} \mathbf{v}_h.
\tag{3.5}
$$

2- Compute the pressure p_h^{n+1}: Find $p_h^{n+1} \in M_h$ such that for all $q_h \in M_h$, we have:

$$\int_\Omega \text{div } \mathbf{u_h^*} q_h + \int_\Omega \frac{\Delta t}{\rho_h^n} \nabla(p_h^{n+1} - p_h^n) \nabla q_h = 0. \tag{3.6}$$

3- Compute the final velocity $\mathbf{u}^{n+1}$: Find $\mathbf{u}_h^{n+1} \in X_{uh}$ such that for all $\mathbf{v}_h \in X_{vh}$, we have:

$$\int_\Omega \varepsilon' \nabla \mathbf{u}_h^{n+1} \nabla \mathbf{v}_h - \int_{\Gamma_1} \beta^{-1} \, {}^t\mathbf{u}_h^{n+1}(\mathbf{n} \quad {}^t\mathbf{n})\mathbf{v}_h ds + \int_\Omega \mathbf{u}_h^{n+1} \mathbf{v}_h = \int_\Omega \mathbf{u}_h^* \mathbf{v}_h - \int_\Omega \frac{\Delta t}{\rho_h^n} \nabla(p_h^{n+1} - p_h^n)\mathbf{v}_h. \tag{3.7}$$

4- Once we obtain $\mathbf{u}_h^{n+1}$, we compute ϕ_h^{n+1} by using the following equation:

$$\int_{\Omega_h} \frac{\phi_h^{n+1} - \phi_h^n oX^n}{\Delta t} r_h + \varepsilon \int_\Omega \nabla \phi_h^{n+1} \nabla r_h - \varepsilon \int_{\Gamma_1} G(\mathbf{u}_h^{n+1}) r_h = 0 \quad \forall r_h \in Y_h, \tag{3.8}$$

It is well known that numerous errors affect the numerical solution in two phase flow modeling. In this algorithm, as time evolves, the algebraic distance property of the Level-Set function is lost and the mass is not preserved. This lack of robustness can be corrected by an algorithm of reinitialization and mass conservation (see for instance [**?**]).

- **Algorithm of reinitialization** We solve iteratively the following equation until it reaches a steady state [**?**], [**?**].

$$\frac{\partial \Phi}{\partial \tau} + w \nabla \Phi = sign(\phi) \quad \text{with } w = sign(\phi) \frac{\nabla \Phi}{\|\nabla \Phi\|}, \tag{3.9}$$

  with

$$\Phi(x, t, \tau = 0) = \phi(x, t), \tag{3.10}$$

  where τ is an imaginary time.

- **Mass conservation** The concept of the method [**?**] is to vary the zero isocontour at each time step by moving the level-set function, i.e. by adding to Φ some signed constant c_Φ, where $|c_\Phi|$ is the distance between the old and new zero-level sets such that the new level-set function Φ_{new} reduces the error of the corresponding mass and defines a new domain

$$\Omega_2^{new} = \{\mathbf{x} \in \Omega : \Phi_{new} > 0\}.$$

  Then the corrected level set function becomes

$$\Phi_{new} \simeq \Phi + c_\Phi \|\nabla \Phi\|. \tag{3.11}$$

By taking into consideration the above reinitialization and mass conservation techniques, the four steps of the above proposed algorithm will be completed by the two following two steps:

4- Reinitialization step:

$$\text{Find } \Phi_h^{n+1} \in Y_h \text{ such that } \forall r_h \in Y_h, \int_\Omega \frac{\Phi_h^{n+1} - \Phi_h^n oX^n_{(\mathbf{w_h}^n, \Delta\tau)}}{\Delta\tau}, r_h - \int_\Omega sign(\phi)_h^n r_h = 0 \tag{3.12}$$

where $\Delta\tau$ is an imaginary time step.

5- Mass conservation step:

$$\Phi_{h_{new}}^{n+1} \simeq \Phi_h^n + c_{\Phi_h^n} \|\nabla \Phi_h^n\|. \tag{3.13}$$

Each step in the above algorithm leads to an algebraic problem to be solved which is defined via a variational formulation on a domain $\Omega \subset \mathbb{R}^d$ for $d \in \mathbb{N}$:

$$\text{Find } \mathbf{u} \in \mathbf{V} \text{ such that} : a_\Omega(\mathbf{u}, \mathbf{v}) = l(\mathbf{v}), \quad \forall \mathbf{v} \in \mathbf{V}, \tag{3.14}$$

where $\mathbf{V}$ is a Hilbert space of functions from Ω with real values.

As we mentioned, the problem is discretized by a finite element method. Let $\mathcal{N}$ denote the set of degrees of freedom and $\mathbf{A} \in M_\mathcal{N}(\mathbb{R})$ be the associated finite element matrix with a generic element $\mathbf{A}_{kl}$.

For some given right-hand side $\mathbf{b}$, we have to solve linear systems of the form:

$$\mathbf{Au} = \mathbf{b}, \tag{3.15}$$

where $\mathbf{u}$ is the desired unknown.

The industrial three dimensional numerical simulations are very expensive in CPU time and memory requirements. In our problem, the projection method reduced the computation time significantly, but it is still impossible to solve problems with large domains on a sequential computer. Thus, we will solve the problem by using a domain decomposition method, the Schwarz method, on a parallel machine.

4. Domain decomposition with Schwarz method

There exists in the litterature different versions of Schwarz algorithms, the Jacobi Schwarz Method (JSM, see [?]), the Additive Schwarz Method (ASM, see [?] and references therein) and the Restricted Additive Schwarz (RAS, see [?], [?]). The first natural feature of these algorithms is that they are equivalent to a Block-Jacobi method when the overlap is minimal. Schwarz methods represent fixed point iterations applied to preconditioned global problems, and consequently they do not provide the fastest convergence possible. Thus, it is natural to apply Krylov methods instead. This provides the justification of using Schwarz methods as preconditioners rather than solvers.

In these methods, the associated mesh of the computational domain Ω is first decomposed into N non-overlapping meshes $\{T_i\}_{1 \leq i \leq N}$ by automatic graph partitioners such as METIS [?] or SCOTCH [?]. Then for a positive integer δ, the overlapping decomposition $\{T_i^\delta\}_{1 \leq i \leq N}$ is defined recursively as follows: T_i^δ is obtained by including all elements of $T_i^{\delta-1}$ plus all its adjacent elements. The number of layers of elements in the overlap is then 2δ.

Let V_h be the finite element space spanned by the finite set of n basis functions $\{\psi_i\}_{1 \leq i \leq n}$ defined on Ω, and let $\{V_{hi}^\delta\}_{1 \leq i \leq N}$ be the local finite element spaces defined on the domains associated to each $\{\omega_i^\delta\}_{1 \leq i \leq N}$, where ω_i^δ is the subdomain defined as the union of all mesh elements in T_i^δ.

Now, consider the restrictions $\{\mathbf{R}_i\}_{1 \leq i \leq N}$ from V_h to $\{V_{hi}^\delta\}_{1 \leq i \leq N}$ and a local partition of unity $\{\mathbf{D}_i\}_{1 \leq i \leq N}$ such that

$$\mathbf{I}_d = \sum_{i=1}^{N} \mathbf{R}_i^T \mathbf{D}_i \mathbf{R}_i.$$

At the algebraic level, if $\{n_i\}_{1 \leq i \leq N}$ denotes the number of degrees of freedom in each local finite element space, then $\mathbf{R}_i$ is a boolean matrix of size $n_i \times n$, and $\mathbf{D}_i$ the diagonal matrix of size $n_i \times n_i$ for all $1 \leq i \leq N$.

The RAS preconditioner [?] is then defined as

$$\mathbf{M}_{RAS}^{-1} = \sum_{i=1}^{N} \mathbf{R}_i^T \mathbf{D}_i \left(\mathbf{R}_i \mathbf{A} \mathbf{R}_i^T \right)^{-1} \mathbf{R}_i.$$

In a parallel environment, the local matrices $\mathbf{A}_i = \mathbf{R}_i \mathbf{A} \mathbf{R}_i^T$ (commonly referred to as "Dirichlet" matrices) are assembled and factorized concurrently on each processor and are also used in the parallel implementation of the global matrix-vector product $\mathbf{Ax}$ for the Krylov method. For more details about the numerical implementation, see [?].

The numerical results presented in this paper were obtained using the finite element software **FreeFem++** [?]. The linear systems are solved with the GMRES [?] algorithm preconditioned by the RAS preconditioner using the parallel library **HPDDM** [?] (High-Performance unified framework for Domain Decomposition Methods), which implements several domain decomposition methods and is interfaced with **FreeFem++**. The results were obtained with an overlap of minimal size ($\delta = 1$). However, the width of the overlap plays a crucial role in the implementation of the method of characteristics as shown in the following.

4.1. **Domain decomposition for the method of characteristics.** In the sequential case (without domain decomposition), it is well known that the characteristic curves may cross some elements of the domain or even go out of the computational domain next to the boundaries (See Fig **??**), especially since the method of characteristics allows us to use a time step greater than the space mesh size. When reaching a domain boundary during the construction of the characteristics, the intersection of the characteristic curve with the boundary is taken as the foot of the characteristic.
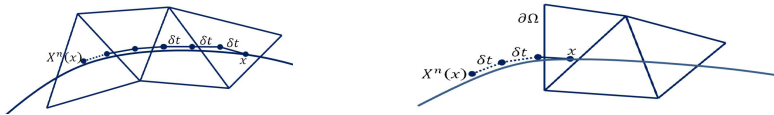
FIGURE 2. (left) traveling characteristic point with time δt inside the domain and (right) characteristic point going out of the domain.

By using the domain decomposition method with minimal overlap $\delta = 1$ (see Figure **??**), if we follow the above technique in each subdomain, the solution will be affected by small numerical errors that accumulate and alter the results as time evolves (as we will see in the numerical results). In order to solve this issue, we propose to extend the subdomain by adding some supplementary overlap layers (see Figure **??**). It is important to note that the extended region is only taken into account for the computation of the characteristic curves that go out of the subdomain (See figure) and s not involved in the domain decomposition algorithm for solving the linear systems. It is obvious that the number of extra overlap layers needed depends on the time step.
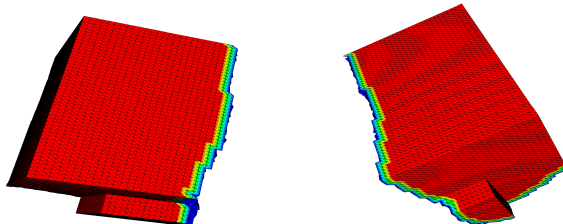

FIGURE 3. Cube subdomains with minimal overlap used in the Schwarz method.
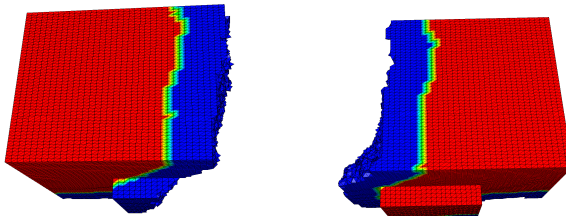

FIGURE 4. Cube subdomains with additional overlap layers for the characteristic curves calculation.

5. NUMERICAL RESULTS

In this section, we perform several numerical simulations using the **FreeFem ++** software [**?**]. We begin with an academic application and we show later industrial cases.

For the numerical investigations, we consider the non-dimensionalized incompressible Navier-Stokes equation:

$$\rho\Big(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}.\nabla\mathbf{u}\Big) - \frac{1}{Re}\mu\,\Delta\mathbf{u} + \nabla p = -\rho_1\frac{1}{Fr^2}\mathbf{e_d}, \tag{5.1}$$

where $Re = \rho_{ref}u_{ref}l_{ref}/\mu_{ref}$ is the Reynolds number, $Fr = u_{ref}/\sqrt{l_{ref}g}$ the Froude number. $\rho_{ref} = \rho_1$ is the density of the first fluid, $\mu_{ref} = \mu_1$ is the viscosity of the first fluid, l_{ref} is the diameter of the inlet and u_{ref} is the average velocity of the fluid.

In order to validate the effectiveness of the method described above, we report the error of the volume evolving in time:

$$err_V = \frac{|V_e - V_n|}{|V_n|},$$

where V_e is the exact volume and V_n is the numerical volume. We compare this error with both the sequential and parallel schemes.

5.1. **First industrial case.** In this section, we model mould filling as considered in [**?**]. The domain Ω is an industrial mould as it is shown in Figure **??**. We intend here to compare the mould filling and
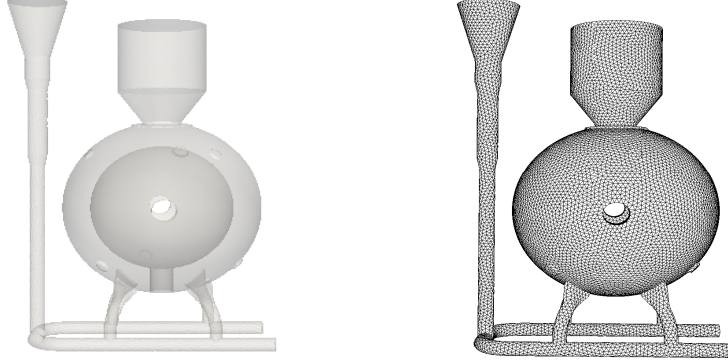
FIGURE 5. To the left: The mould shape; To the right: the mould mesh.

the interface transport that separates the two fluids in the sequential code and the parallel code. In this simulation, The density of air is $1\,\mathrm{Kg\,m^{-3}}$, the kinematic viscosity is set to be $1.818 \times 10^{-3}\,\mathrm{m^2\,s^{-1}}$ and the dynamic viscosity is set to be $1.818 \times 10^{-3}\,\mathrm{Kg\,m^{-1}\,s^{-1}}$; whereas the density of the aluminium liquid is $2385\,\mathrm{Kg\,m^{-3}}$, the kinematic viscosity is set to be $10^{-4}\,\mathrm{m^2\,s^{-1}}$ and the dynamic viscosity is set to be $2.385 \times 10^{-5}\,\mathrm{Kg\,m^{-1}\,s^{-1}}$. The entry surface is $0.045\,\mathrm{m^2}$.

The Reynolds number is $Re = 10\,000$. Furthermore, we take $U_{\mathrm{in}} = 0.5\ m/s$, $h = 0.02$, $\Delta t = 2h$ and $\varepsilon = h/5$. The considered mesh contains 59364 vertices and 294544 tetrahedrons. At the initial time, the interface Γ is represented by the Level-Set function of equation $\phi_0 = -z + 0.9$.

Figures **??** shows the evolution of the interface at the initial time as well as at $t = 0.8\ s$, $t = 2\ s$ and $t = 7.2\ s$. Figure **??** shows a comparison of the evolution of the volume error with respect to
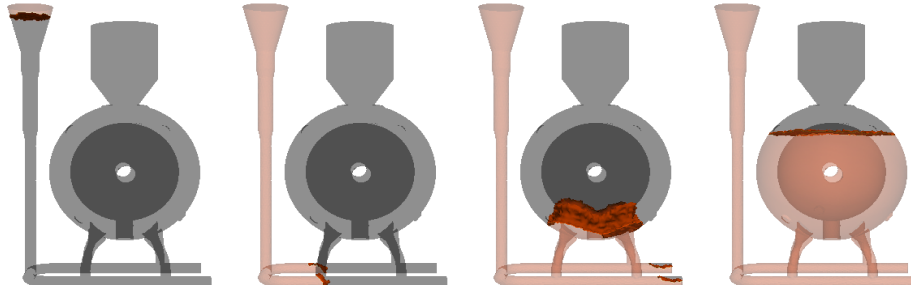

FIGURE 6. The approximated interface at the initial time, $t = 0.8\ s$, $t = 2\ s$ and $t = 7.2\ s$.

time without applying the mass conservation algorithm, with the sequential code, the parallel code with minimal overlap,and the parallel code with additional overlap layers for the method of characteristics. We can see that with the addition of the supplementary layers of overlap, we recover the correct results with the parallel code.
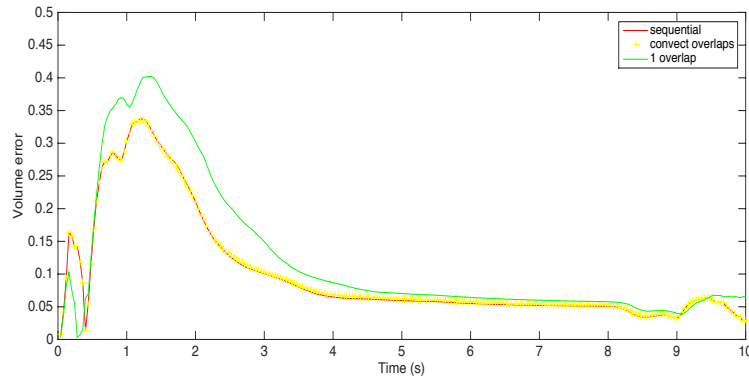
FIGURE 7. The evolution of the volume error with respect to time for $dt = 2h$ with the sequential code, the parallel code with one overlap and the parallel code with additional overlap layers (convect overlaps).

In Figure **??** we present the volume error after applying the mass conservation algorithm. We can see the large effect of the mass conservation algorithm on the error, with a maximal error of 0.0034 (versus 0.4 without the mass conservation step). Figure **??** shows the computation time of the simulation in the
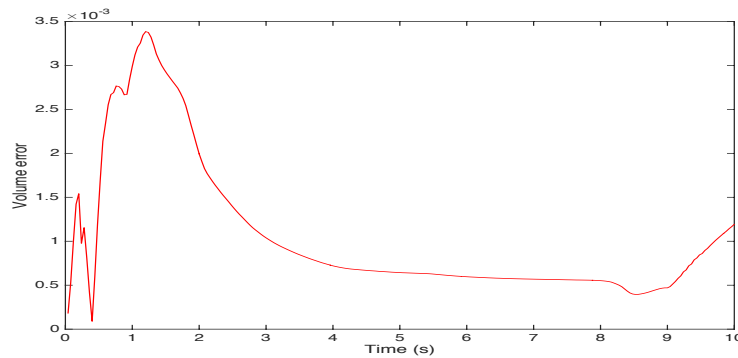
FIGURE 8. Variation of the volume error with the mass conservation step for the first industrial case.

first industrial case. At time $t = 10s$, the computation time decreases from 42 hours in the sequential code to approximatively 1.4 hours with the parallel code using 128 cores.
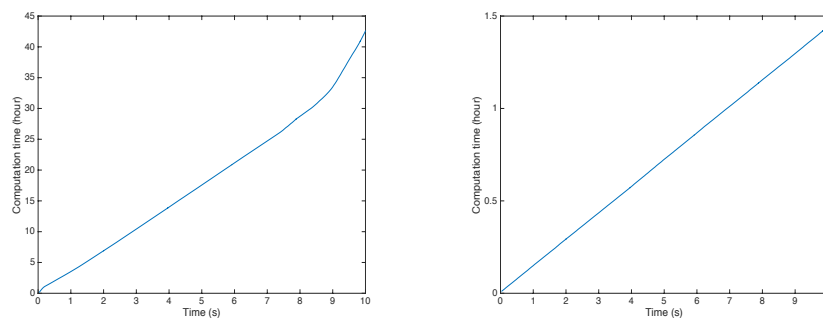

FIGURE 9. computation time of the simulation in the first industrial case. To the left: sequential code; To the right : parallel code.

Figure **??** illustrates a strong scalability test and gives the computation time in logarithmic scale for the 5 matrices involved in the first iteration (corresponding to the 5 steps of the algorithm presented in

Section **??**) with an increasing number of cores, from 1 to 128. We can see that we obtain a speedup of approximately 30 when going from 1 to 128 cores. Moreover, we observe a decrease in efficiency with the number of cores, with a speedup of 2 when going from 1 to 2 cores, to a speedup of approximately 1.3 when going from 64 to 128 cores. This decrease in efficiency depends in fact on the size of the global problem (in terms of degrees of freedom). In what follows, we will use the parallel algorithm for another mould of larger size, where the problem cannot be solved with the sequential code due to CPU and memory limitations.
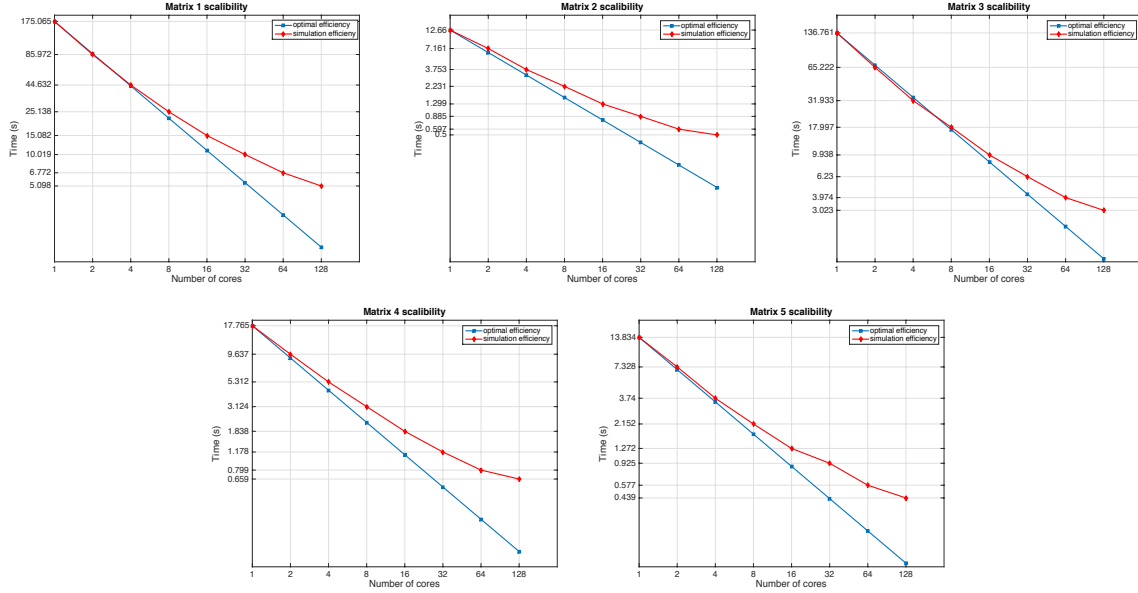

FIGURE 10. Strong scalability test for the industrial problem. By order, matrix 1 to 5 from up to down and left to right.

5.2. **Second industrial case.** In this section, we model the filling of another industrial mould with a larger size, the maximal height of the considered mould is $1.7\ m$; whereas its largest diameter is $1.56\ m$. The geometry of this mould is given by Figure **??**. The density of air is $1Kg.m^{-3}$, its cine-
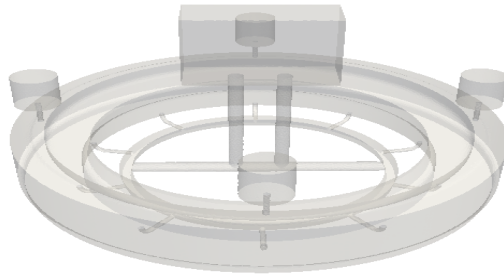

FIGURE 11. The second indutrial mould.

matic viscosity is $1.818e^{-3}m^2.s^{-1}$ and its dynamic viscosity is $1.818e^{-3}Kg.m^{-1}.s^{-1}$; whereas the density of liquid aluminium is $2385Kg.m^{-3}$, its cinematic viscosity is $1e^{-4}m^2.s^{-1}$ and its dynamic viscosity is $2.385e^{-5}Kg.m^{-1}.s^{-1}$. In addition, $\text{U}_{\text{in}} = 0.25\ m/s$, $h = 0.02$ and $\Delta t = 3h$.
The considered mesh contains $2,646,497$ tetrahedrons and $468,362$ vertices.

Figures **??** and **??** show the evolution of the approximated interface at $t = 0\ s$, $t = 3\ s$, $t = 6\ s$, $t = 9\ s$, $t = 12\ s$ and $t = 15\ s$.
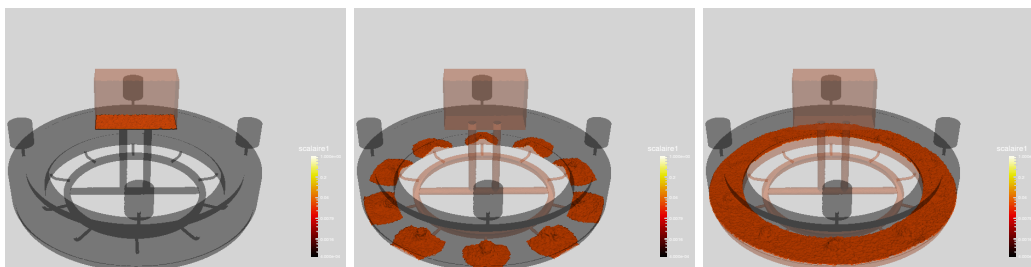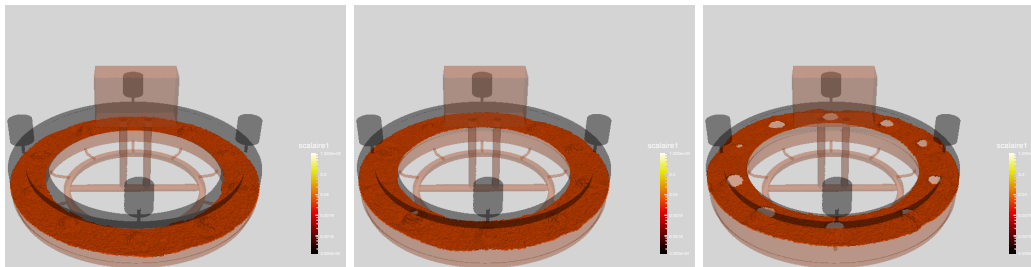
FIGURE 12. The approximated interface at t=0 s, t=3 s and t=6 s

FIGURE 13. The approximated interface at t=9 s, t=12 s and t=15 s

Figure ?? shows the volume error variation in time during the mould filling. Numerical difficulties were encountered in performing the mass conservation in the pipes (for $t < 6s$), the volume error is large in the pipes at each time step. This forbade us from applying the mass conservation algorithm until $t \geq 6$ where we get small errors of volume.
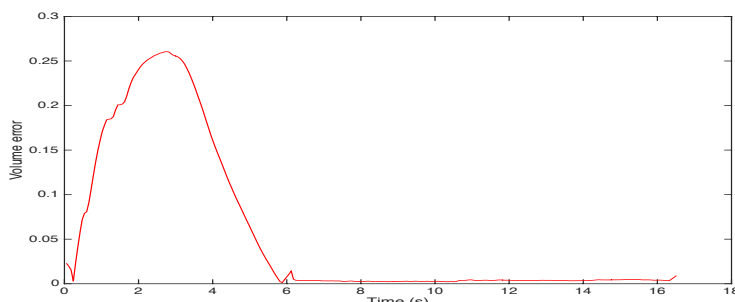

FIGURE 14. Error variation in terms of time.

In figure ??, we present the computation time in logarithmic scale for the 5 matrices involved in the first iteration for an increasing number of cores, from 2 to 128. We present the strong scalability results for the first iteration in time because the GMRES convergence is the same regardless of what is happening with respect to the physics.

When going from 2 to 128 cores we obtain a speedup of 52 for the first matrix, 47 for the second, 104 for the third, 37 for the fourth and 46 for the fifth. The average speedup is 1.9 when going from 2 to 4 cores, but decreases to 1.7 when going from 64 to 128 cores.

Figure ?? shows the computation time for the parallel simulation using 128 cores. The simulation takes only about 5 hours.

Even if we obtain better speedups in the second industrial case compared to the smaller first case, we still observe a decrease in parallel efficiency for an increasing number of cores. This is a well-known characteristic of one-level overlapping domain decomposition preconditioners such as RAS, for two reasons: First,
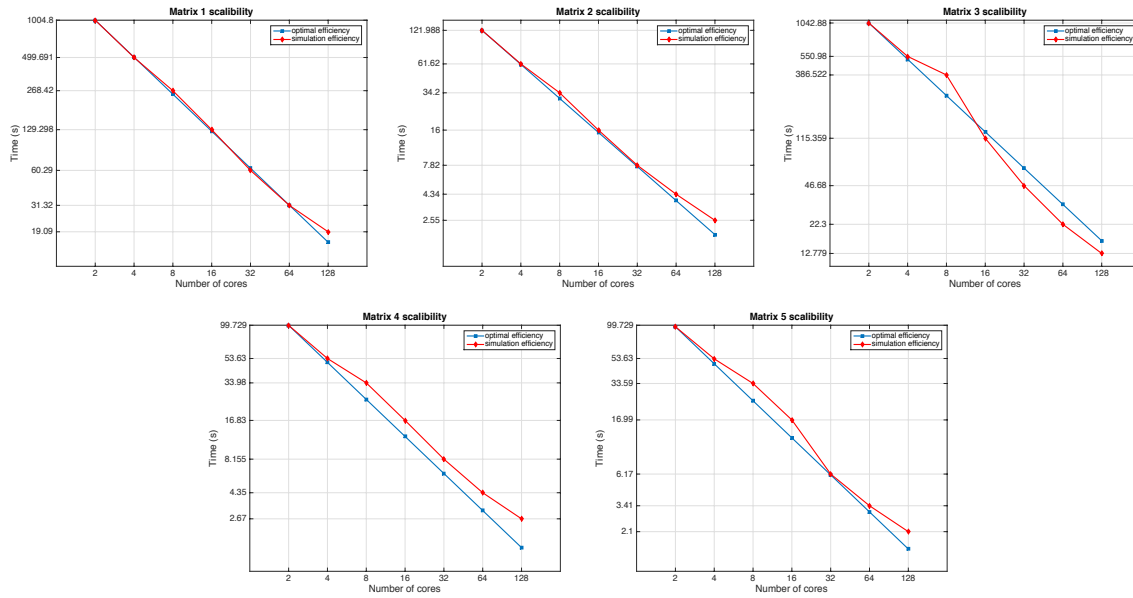
FIGURE 15. Strong scalability test for the second industrial problem. By order, matrix 1 to 5 from up to down and left to right.
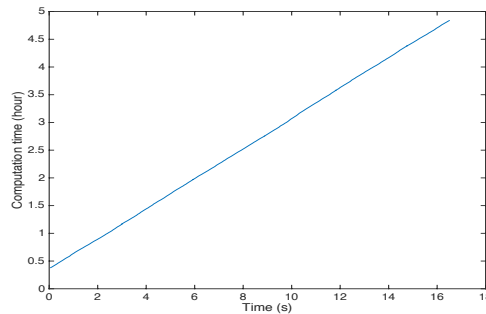

FIGURE 16. Computation time for the simulation of the second industrial case with the parallel code.

overlapping methods naturally lose efficiency when the subdomain size becomes too small. Second, one-level methods only rely on local exchanges between neighbouring subdomains, with no global coupling. One can show that with such methods the condition number of the preconditioned system depends on the number of subdomains. To remedy this problem, two-level preconditioners have been introduced [**?**] that enrich the one-level method with a so-called coarse operator, providing global transfer of numerical information across all subproblems. Future work will consist in using two-level domain decomposition preconditioners to obtain a highly scalable parallel algorithm for solving problems with very large moulds on thousands of cores.

References

[1] M. Haddad, F. Hecht and T. Sayah. , Interface transport scheme of a two-phase flow by the method of characteristics, *International Journal for Numerical Methods in Fluids, DOI: 10.1002/fld.4280, August 2016*

[2] Batchelor G.K. , An Introduction to Fluid Dynamics, *Cambridge University Press, 1967.*

[3] Lamb H., Hydrodynamics., *Cambridge University Press, 1932.*

[4] Navier C.L.M.H., Sur les lois de l'équilibre et du mouvement des corps élastiques, *Mem. Acad. R. Sci. Inst. Vol. 6, France (1827) 369.*

[5] Babuska I., The Finite Element Method with Penalty , *Mathematics of computation, Volume 27, Number 122, April 1973.*

[6] Chorin A.J., Numerical solution of the Navier-Stokes equations, *Math. Comp. 22 (1968) 745-762* .

[7] CHORIN A.J., On the convergence of discrete approximations to the Navier- Stokes equations, *Math. Comp. 23 (1969) 341-353* .

[8] SUSSMAN M., SMEREKA P., OSHER S., A level set approach for computing solutions to incompressible two-phase flows, *J. Comput. Phys., 1994; 114: 146-159.*

[9] PENG D., MERRIMAN B., OSHER S., ZHAO H., AND KANG M., A PDE-based fast local level set method, *Journal of Computational Physics, 155, pp. 410-438, 1999.*

[10] A. SMALONSKI , Numerical Modeling of Two-Fluid Interfacial Flows , *PhD thesis University of Jyvaskyla Finland 2001.*

[11] DERVIEUX A., THOMASSET F., A finite element method for the simulation of a Rayleigh-Taylor instablity, in Approximation Methods for Navier-Stokes problems, *Lecture Notes in Mathematics, 1980; 771, Springer- Verlag, Berlin, 145-158.*

[12] BENQUÉ J.P., IBLER B., KERAMSI A., LABADIE G., A finite element method for the Navier- Stokes equations, Proceedings of the third international conference on finite elements in flow problems , *Banff. Alberta, Canada 10-13 June (1980).*

[13] H. A. SCHWARZ, Uber einen Grenzubergang durch alternierendes Verfahren. , *Vierteljahresschrift der Naturforschenden Gesellschaft in Zurich, 15:272-286, 1870.*

[14] MARTIN J. GANDER, Schwarz methods over the course of time, *Electronic transactions on numerical analysis, 31:228-255, 2008* .

[15] GEORGE KARYPIS AND VIPIN KUMAR. METIS, unstructured graph partitioning and sparse matrix ordering system. version 2.0. Technical report , *Minneapolis, MN 55455: University of Minnesota, Department of Computer Science, August 1995.*

[16] C. CHEVALIER AND F. PELLEGRINI. PT-SCOTCH, a tool for efficient parallel graph ordering, *Parallel computing, 6-8(34):318-331, 2008* .

[17] VICTORITA DOLEAN, PIERRE JOLIVET, FREDERIC NATAF, An Introduction to Domain Decomposition Methods: algorithms, theory and parallel implementation, *Master. France. 2015.*

[18] ANDREA TOSELLI AND OLOF WIDLUND, Domain decomposition methods -algorithms and theory, *Volume 34 of Springer Series in Computational Mathematics. Springer, 2005.*

[19] XIAO-CHUAN CAI, CHARBEL FARHAT, AND MARCUS SARKIS, A minimum overlap restricted additive Schwarz preconditioner and applications to 3D flow simulations, *Contemporary mathematics, 218:479-485,1998.*

[20] XIAO-CHUAN CAI AND MARCUS SARKIS, A restricted additive Schwarz preconditioner for general sparse linear systems, *Siam journal on scientific computing, 21:239-247, 1999.*

[21] PIERRE JOLIVET, FRÉDÉRIC HECHT, FRÉDÉRIC NATAF AND CHRISTOPHE PRUD?HOMME, Scalable Domain Decomposition Preconditioners For Heterogeneous Elliptic Problems, *Proceeding SC'13 Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis Article No. 80.*

[22] HECHT F., New development in FreeFem++, *J. Numer. Math. 20 (2012), no. 3-4, 251-265. 65Y15.*

[23] VICTORITA DOLEAN, PIERRE JOLIVET, FREDERIC NATAF, NAn Introduction to Domain Decomposition Methods: algorithms, theory and parallel implementation, *J. Master. France. 2015.*

[24] Y. SAAD AND M. SCHULTZ, A generalized minimum residual algorithm for solving nonsymmetric linear systems, *SIAM J. Sci. Stat. Comput., 7 (1986), pp. 856?869.*

[25] T. A. DAVIS, AND I. S. DUFF, A Combined Unifrontal/Multifrontal Method for Unsymmetric Sparse Matrices, *ACM Trans. Math. Softw., Vol. 25, No. 1, 1997, pp. 1?19.*

[26] P. K. GUPTA AND K. V. PAGALTHIVARTHI, Application of Multifrontal and GMRES Solvers for Multisize Particulate Flow in Rotating Channels, *Prog. Comput Fluid Dynam., Vol. 7, 2007, pp. 323?336.*

[27] M. P. RAJU, Parallel Computation of Finite Element Navier- Stokes Codes Using MUMPS Solver, *International Journal of Computer Science Issues, Vol. 4, No. 2, Sept. 2009.*

[28] P. R. AMESTOY, I. S. DUFF AND J.-Y. L'EXCELLENT, Multifrontal Parallel Distributed Symmetric and Unsymmetric Solvers, *Comput. Methods in Appl. Mech. Eng., Vol. 184, 2000, pp. 501-520.*

[29] P. R. AMESTOY, I. S. DUFF, J. KOSTER AND J.-Y. L'EXCELLENT, A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling, *SIAM Journal of Matrix Analysis and Applications, Vol. 23, No. 1, 2001, pp. 15-41.*

[30] P. R. AMESTOY, A. GUERMOUCHE, J.-Y. L'EXCELLENT AND S. PRALET, Hybrid Scheduling for the Parallel Solution of Linear Systems, *Parallel Computing, Vol. 32, No. 2, 2006, pp. 136-156.*

[31] Y. SAAD, Iterative Methods for Sparse Linear Systems, *SIAM Publishers, 2nd edition, 2000.*

[32] A.TOSELLI AND O. B. WIDLUND, Domain decomposition methods: algorithms and theory, *Vol. 34. Berlin: Springer, 2005.*