Parallel Simulation of Multiphase/Multicomponent Flow Models

Erlend Øian, Magne S. Espedal, I. Garrido, and G. E. Fladmark

University of Bergen, Center for Integrated Petroleum Research (CIPR)
http://www.mi.uib.no/~erlendo/

Summary. The simulation of flow in porous media is a computationally demanding task. Thermodynamical equilibrium calculations and complex, heterogeneous geological structures normally gives a multiphysics/multidomain problem to solve. Thus, efficient solution methods are needed. The research simulator ATHENA is a 3D, multiphase, multicomponent, porous media flow simulator. A parallel version of the simulator was developed based on a non-overlapping domain decomposition strategy, where the domains are defined a-priori from e.g. geological data. Selected domains are refined with locally matching grids, giving a globally non-matching, unstructured grid. In addition to the space domain, novel algorithms for parallel processing in time based on a predictor-corrector strategy has been successfully implemented.

We discuss how the domain decomposition framework can be used to include different physical and numerical models in selected sub-domains. Also we comment on how the two-level solver relates to multiphase upscaling techniques.

Adding communication functionality enables the original serial version to run on each sub-domain in parallel. Motivated by the need for larger time steps, an implicit formulation of the mass transport equations has been formulated and implemented in the existing parallel framework. Further, as the Message Passing Interface (MPI) is used for communication, the simulator is highly portable. Through benchmark experiments, we test the new formulation on platforms ranging from commercial super-computers to heterogeneous networks of workstations.

1 Introduction

The simulation of flow in porous media is a computationally demanding task. Thermodynamical equilibrium calculations and complex, heterogeneous geological structures normally gives a multiphysics/multidomain problem to solve. When studying e.g. various faulted and fractured porous media, important features that can have a large impact on the flow characteristics are localized in space and exist on a much smaller scale than the characteristic length scale of the domain of interest.

In order to give a full three dimensional description of geometrical and physical properties of such a case, efficient numerical tools are necessary. A natural approach to resolve the geometrical details, are local grid refinement (LGR) techniques. The goal of such methods is to reduce the overall size of the problem while retaining a fairly good numerical resolution. A domain decomposition based LGR technique was implemented in an in-house, 3D, research simulator for porous media flow called ATHENA, see Reme and Øye [1999]. By adding communication functionality, the original serial version was extended to run on each sub-domain in parallel, see Øye and Reme [1999]. The communication is enabled through an object oriented, C++ library called OOMPI. This library is based on the Message Passing Interface (MPI) standard.

The framework included in the ATHENA simulator allows various aspects of domain decomposition strategies to be explored. In the space domain different models and discretizations can be used within the total domain. In a similar way, the time domain can be split and solved in parallel. This is achieved through a predictor-corrector strategy in which a coarse time step simulation (predictor) provides initial values for solving fine sub-intervals in parallel (corrector).

As an application, we will show how the domain decomposition framework can be used for modeling flow in fractured porous media. Specifically, we suggest applying a discrete fracture network model in selected domains. Such a model is a flexible and accurate tool to describe the complex geometries of fractures, but at the cost of larger systems of equations. This problem can be solved by using parallel computations and upscaling.

The mathematical model describing multiphase porous media flow includes equations for the mass transport. Previously, these equations were solved by a forward Euler time stepping scheme. Motivated by the need for larger time steps, an implicit formulation of the mass transport equations was formulated. Here we will describe how the implicit formulation is included in the framework of a parallel version of the ATHENA simulator.

We compare the implicit formulation on platforms ranging from commercial super-computers to heterogeneous networks of PC workstations.

In Sect. 2 we recall the mathematical model of porous media flow with multiple phases and thermal effects. Then, in Sect. 3, we present the domain decomposition and local grid refinement framework. The approach which combines fracture modeling and domain decomposition is given in Sect. 4 and a parallel implementation of implicit mass transport formulation in Sect. 5. Section 6 includes an example that combines aspects of the framework presented above. We end with a summary and conclusion in Sect. 7.

2 Mathematical Model

The mathematical model describing multiphase flow in porous media with multiple components and thermal effects constitutes a complex set of coupled

equations. These equations involve a set of primary variables and additional constraints imposed by secondary variables. Since the model we use has already been described in detail in e.g. Reme et al. [2000], we will only briefly present the equations for the primary variables. The $2+n_c$ primary variables are the *temperature*, T, the *water pressure*, p^w, and the *molar masses*, N_ν, of each component. Here $\nu = 1, 2, \ldots, n_c$, is the component index, and n_c is the number of components. Further, we let V denote a finite control volume of a porous medium having the closed surface, S. In order to simplify notation we leave the summation index unspecified when summing over phases, i.e.

$$\sum_\ell \equiv \sum_{\ell=\mathrm{w,o,g}} .$$

The temperature within a control volume, V, is governed by a heat flow equation. This equation expresses conservation of energy by relating the temperature gradient, ∇T, the heat capacity, ρu, convective flux, $h\rho\mathbf{u}$, and heat sinks/sources q:

$$\frac{\partial}{\partial t}\int_{\mathrm{V}} (\rho u) dV - \int_{\mathrm{S}} \left(\underline{k}\nabla T - h\rho\mathbf{u}\right)\cdot d\mathbf{S} = \int_{\mathrm{V}} q dV. \tag{1}$$

An equation for the water pressure is derived by requiring that the pores are totally filled, i.e. that the residual pore volume, $R(t) = 0$, $\forall t$. A Taylor's expansion of $R(t + \Delta t)$ then gives:

$$\frac{\partial R}{\partial p^w}\frac{\partial p^w}{\partial t} + \sum_{\nu=1}^{n_c} \frac{\partial R}{\partial N_\nu}\frac{\partial N_\nu}{\partial t} = -\frac{R}{\Delta t} - \frac{\partial R}{\partial W}\frac{\partial W}{\partial t}. \tag{2}$$

The overburden pressure $W = \sigma + p$ is the sum of effective stress, σ, and pore pressure, p. This derivation of the pressure equation gives a sequential formulation of the mathematical model.

Finally, we have n_c equations expressing conservation of the molar mass of component ν:

$$\frac{\partial}{\partial t}\int_{\mathrm{V}} \Big(\phi_p \sum_\ell C_\nu^\ell \xi^\ell S^\ell\Big) dV = -\int_{\mathrm{S}} \Big(\sum_\ell C_\nu^\ell \xi^\ell \mathbf{v}^\ell\Big)\cdot d\mathbf{S} + \int_{\mathrm{V}} q_\nu dV. \tag{3}$$

Here, ϕ_p is the rock porosity, C_ν^ℓ is the fraction of component ν in phase ℓ, and ξ^ℓ, S^ℓ and $\mathbf{v}^\ell$ are the corresponding molar density, phase saturation and generalized Darcy velocity respectively.

In order to solve these equations numerically, we use a standard, cell centered, piecewise constant finite volume discretization in space. Details on how the resulting systems of equations are solved in each time step are given in the next section.

3 A Two-level Solver

Here we present an iterative solver which we use to solve the linear systems. The solver is equivalent to the so called Fast Adaptive Composite method, see e.g. Teigland [1998], Briggs et al. [2000]. It can also be viewed as a two-level domain decomposition method, see Smith et al. [1996] and references therein.

Assume that a matching, but possibly non-regular, *coarse grid*, $\hat{\Omega}$ is defined. Further, let a subset, Ω_f, of coarse cells be refined. Each of the coarse cells in this subset defines a sub-domain, Ω_{f_i}, that is refined independently of the other sub-domains with a locally matching, non-regular grid. The resulting *composite grid*, Ω, is generally non-regular and non-matching and consists of some (or no) true coarse cells, Ω_c, and several (or all) refined sub-domains

$$\Omega_f = \bigcup_{i=1}^{p} \Omega_{f_i}, \tag{4}$$

where p is the number of refined sub-domains.

Let the number of composite grid cells in the composite grid be N. The underlying coarse grid has $\hat{N}$ cells. The *index set* of fine cells within coarse cell number $\hat{i}$ is denoted $\mathrm{M}_{\hat{i}}$. Further, we associate to this cell a *basis vector* $\boldsymbol{\psi}_{\hat{i}} \in \mathbb{R}^N$ defined as

$$\boldsymbol{\psi}_{\hat{i}} = \{\psi_k^{\hat{i}}\}. \tag{5}$$

The vector components, $\psi_k^{\hat{i}}$, have value one for refinement cell, k, in coarse cell/sub-domain number $\hat{i}$, i.e.

$$\psi_k^{\hat{i}} = \begin{cases} 1, & \forall\, k \in \mathrm{M}_{\hat{i}}, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

Then, let $\mathcal{R}^T \in \mathbb{R}^{N \times \hat{N}}$ be the matrix representation of interpolation from the coarse grid, $\hat{\Omega}$, to the composite grid, Ω. The columns of this operator consist of the basis vectors $\boldsymbol{\psi}_{\hat{i}}$. Thus representing constant interpolation. Correspondingly, the restriction operator is $\mathcal{R} \in \mathbb{R}^{\hat{N} \times N}$. Further, the restriction operator $\mathcal{S}_i \in \mathbb{R}^{N_i \times N}$, returns the vector coefficients defined on sub-domain Ω_i, i.e.

$$\mathbf{x}_{\Omega_i} = \mathcal{S}_i \mathbf{x}, \qquad \mathbf{x}_{\Omega_i} \in \mathbb{R}^{N_i}, \quad \mathbf{x} \in \mathbb{R}^N. \tag{7}$$

Finally, the combination of $\mathcal{R}^T$ and $\mathcal{S}_i$ provides a mapping $\mathcal{R}_i^T \in \mathbb{R}^{N_i \times \hat{N}}$ from the coarse grid to sub-domain Ω_i:

$$\mathcal{R}_i^T = \mathcal{S}_i \mathcal{R}^T. \tag{8}$$

The numerical solution of the equations in the preceding section entails solving linear systems of the form $\mathcal{A}\mathbf{x} = \mathbf{b}$. This can be written in block matrix form,

$$\begin{bmatrix} \mathcal{A}_{cc} & \mathcal{A}_{cf} \\ \mathcal{A}_{fc} & \mathcal{A}_{ff} \end{bmatrix} \begin{bmatrix} \mathbf{x}_c \\ \mathbf{x}_f \end{bmatrix} = \begin{bmatrix} \mathbf{b}_c \\ \mathbf{b}_f \end{bmatrix} \tag{9}$$

where the system matrix is decomposed according to the domains of the grid. We use the various transfer operators defined above to define a two-level solution algorithm for the system in Eq. (9) defined on the composite grid.

We proceed by introducing the two-level iterated solution $\mathbf{x}^{(s)}$ of Eq. (9). Let an updated/improved solution be defined by

$$\mathbf{x}^{(s+1/2)} = \mathbf{x}^{(s)} + \mathcal{R}^T \hat{\mathbf{d}}^{(s)}, \tag{10}$$

where $\hat{\mathbf{d}}^{(s)} \in \mathbb{R}^{\hat{N}}$ is a coarse grid correction. Substituting $\mathbf{x}^{(s)}$ for the update $\mathbf{x}^{(s+1/2)}$ and restricting to the coarse grid, we get

$$\mathcal{R}\mathcal{A}\mathcal{R}^T \hat{\mathbf{d}}^{(s)} = \mathcal{R}\big(\mathbf{b} - \mathcal{A}\mathbf{x}^{(s)}\big). \tag{11}$$

The solution of this equation gives the next step, iterated solution for the non-refined coarse cells, and an intermediate update of the refined sub-domain solutions defined by

$$\mathbf{x}_c^{(s+1)} = \mathbf{x}_c^{(s+1/2)} = \mathbf{x}_c^{(s)} + \mathcal{R}_c^T \hat{\mathbf{d}}^{(s)}, \tag{12}$$

$$\mathbf{x}_r^{(s+1/2)} = \mathbf{x}_r^{(s)} + \mathcal{R}_r^T \hat{\mathbf{d}}^{(s)}, \quad r = 1, \ldots, p. \tag{13}$$

The intermediate solutions $\mathbf{x}_r^{(s+1/2)}$ enable the composite problem in Eq. (9) to be split into independent sub-domain problems for all the sub-domains $q = 1, \ldots, p$

$$\mathcal{A}_{qq}\mathbf{x}_q^{(s+1)} = \mathbf{b}_q - \sum_{r \neq q} \mathcal{A}_{qr}\mathbf{x}_r^{(s+1/2)}, \tag{14}$$

where summation index r also includes the set of true coarse cells. The iteration proceeds until the scaled difference between two consecutive iterations is below some prescribed tolerance.

Our group is also working on applying the two-level scheme above to the time domain, following a parallel technique proposed in Baffico et al. [2002]. A coarse time step solution acts as a predictor by providing boundary values for each sub time interval of the coarse step. Then, each sub time step problem is solved, determining a correction to the coarse solution for the next step of the iteration, see Garrido et al..

4 Using the Domain Decomposition Framework

The domain decomposition based local grid refinement strategy we have implemented is similar to what is know as multiblock reservoir simulation, see e.g. Lee et al. [2002] and Lu et al. [2002]. Multiblock grids allows different

gridding in each block, i.e. coarse domain, with possibly non-matching gridlines at block boundaries, Lee et al. [2003]. The two-level solver for composite grids in Sect. 3 can be described as a multiblock method. An important advantage of this approach is that it combines the simplicity of globally structured grids with the flexibility of fully unstructured discretizations. Further, since the coarse (Galerkin) operator of Eq. (11) is defined simply as a summation of the fine scale operators, it can be constructed independently of the type of mathematical model which is used on each sub-domain.

A basic requirement in the cell centered finite volume method, is that flux is continuous across interfaces of the grid. In our case, we want the mass to be conserved, i.e. we want continuity of mass flux. For composite interfaces, we introduce ghost cells and calculate the composite interface fluxes by interpolating real cell pressure (potential) values onto the ghost cells. Currently, we only use a constant interpolation.

The so called mortar methods provide a general framework that treats composite interfaces in a systematic way, see e.g. Ewing et al. [2000] et al. for a finite volume element variant and references therein.

Due to our relatively crude composite interface approximation, we currently require that the coarse block interfaces are located away from large gradients and boundaries between regions of different physical properties/models. This implies that the transition zone between e.g. a single phase and a multiphase flow region should be included within a multiphase model coarse block. Another case is flow through faulted porous media. We use such an example to illustrate the use of our framework in Sect. 6 below.

4.1 Discrete Fracture Network Model

As an application of the multiblock/multiphysics and parallel processing capabilities of our simulator, we consider flow in fractured porous media. In particular, we are working on combining structured, Cartesian discretizations with discrete network models in selected domains, Øian [2004]. This applies to fractured porous media simulations, where the fractures occur in localized swarms and the traditional dual continuum approach might not be appropriate. In Karimi-Fard et al. [2003] and Karimi-Fard and Firoozabadi [2003] a discrete fracture model is presented. It is based on unstructured grids and allows for both two and three-dimensional systems. An important aspect of this method is that control volumes at the intersection of fractures can be removed, thus relaxing the restrictions on the stable time step in the simulations. Due to the flexibility of this method in modeling complex geometries, fine scale effects are resolved accurately. This method might easily introduce too many details, though, if it is used on the global domain. The multiblock framework enables us to localize this method to selected domains based on the geological description. In the rest of the domain, we can use traditional, less expensive discretizations. This is illustrated in Fig. 1, where only the upper right domain is discretized with the discrete fracture network model.

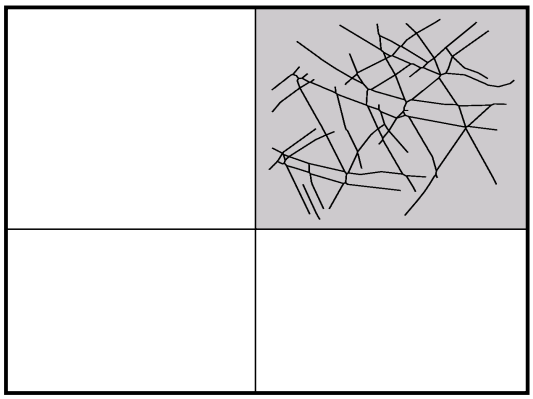

Fig. 1. The figure illustrates using a discrete fracture model in a single domain.

4.2 Upscaling Issues

Since geological models of porous media are highly detailed, direct simulations on such models are typically not efficient or feasible. The standard approach to tackle this scale discrepancy problem, is to use various methods of coarsening or upscaling. For single phase flow simulations it is common to calculate an upscaled, effective absolute permeability through either analytical averaging techniques or local numerical methods. Such (quasi-)local methods, may not be adequate in many situations. Typically, these methods can be sensitive to the boundary conditions used. In contrast, global methods use full fine scale simulations to determine the coarse scale parameters and are thus better at capturing coarse scale flow features, but at the expense of more computations. These methods all focus on upscaling of the permeability. Other variants calculate transmissibilities directly.

When we turn to multiphase flow problems, the effect of relative permeabilities and capillary pressures must also be considered. It is common to upscale the various parameters independently. The problem, though, is that combining these upscaled parameters might not give a coarse model which captures the coarse scale features of the flow. Several authors have suggested that due to the complexity and uncertainty in such an approach, upscaling of the solution variables directly should be considered.

The coarse solver found in many preconditioning techniques/multilevel iterative methods directly incorporates all the fine scale information. Multigrid-upscaling methods are based on the idea/observation that these coarse operators might be good approximations of the coarse scale effect of the fine scale differential operator. Various work on upscaling within a multigrid context are given in e.g. Moulton et al. [1998] and Knapek [1998, 1999].

The two level multigrid solver that is implemented in the ATHENA simulator fits this framework. In Aarnes et al. and Reme et al. [2002] a Galerkin-based upscaling procedure was presented. The main idea is to use existing

information on the fine scale in the coarse averaging system, i.e. to capture some of the fine grid flow internal to each coarse grid block without solving the full fine grid problem. It is important to note that by using piecewise constant interpolation we ensure that mass is conserved. By solving an inverse problem based on the coarse solution, effective parameters can be calculated. In the single phase case, using Darcy's law on the coarse scale with volume averaged velocities and pressures provides a set of equations for the components of the permeability tensor.

Based on some upscaled parameters, a coarse solve is performed. The coarse solution provides boundary conditions for local fine scale simulations in order to determine domains for further refinement. As this process is only performed once (or a few times) during the course of a simulation, the overhead of the fine scale simulations is permissible. We propose an algorithm which includes both a local simulation approach and multigrid upscaling. The local solve includes improved boundary conditions stemming form a global coarse solve. This is similar to the coupled local-global approach in Chen et al. [2003]. This could serve as a background permeability for a full multiphase simulation. But, more research is needed in order to fully understand the implications of such an approach.

5 Implicit Molar Mass Formulation

Discretizing e.g. fractured domains using locally refined grids, leads to small spatial scales. Previously the mass transport equations were integrated using an explicit, forward Euler scheme. To avoid the severe time step restrictions given by the CFL-condition, an implicit formulation using a backward Euler scheme has been formulated, see Chaib et al. [2002], Øian et al. [2003], Garrido et al..

We start by noting that the molar mass of component ν is equal to the integral on the left hand side of Eq. (3). We let V^ℓ denote the volume of phase ℓ and introduce the volume factor $a^\ell = 1/V^\ell$. Further, we define the molar mass of component ν in phase ℓ as $N_\nu^\ell = C_\nu^\ell N^\ell = C_\nu^\ell V^\ell \xi^\ell$. Then, integrating Eq. (3) over a control volume V_i, with surface S_i in a numerical grid and using up-stream weighting, we get

$$\frac{\partial N_{\nu_i}}{\partial t} + \sum_{\mathrm{is}\in \mathrm{S}_i} \left(\sum_\ell \left(a^\ell N_\nu^\ell\right)_{\mathrm{in}} \theta_{\mathrm{is}}^\ell \right) = Q_{\nu_i}, \tag{15}$$

where $\theta_{\mathrm{is}}^\ell = \mathbf{v}_{\mathrm{is}}^\ell \cdot \mathbf{n}_{\mathrm{is}} A_{\mathrm{is}}$ is the Darcy volume flux. Here $\mathbf{n}_{\mathrm{is}}$ and A_{is} is the outward normal and area of subsurface "is" of S_i respectively. Subscript "in" indicates evaluation in the upstream cell and is phase dependent.

After a Newton-Raphson linearization step, we get

$$\frac{\delta N_{\nu_i}^{(t+1)}}{\Delta t} + \sum_{\mathrm{is}\in \mathrm{S}_i} \sum_\ell \left[\left(a^{\ell^n} \sum_\mu \left(\frac{\partial N_\nu^\ell}{\partial N_\mu} \right)^{(t)} \delta N_\mu^{(t+1)} \right)_{\mathrm{in}} \theta_{\mathrm{is}}^{\ell^n} \right] = \beta_{\nu_i}^{(t)}. \tag{16}$$

The right hand side is given by

$$\beta_{\nu_i}^{(t)} = Q_{\nu_i}^n - \frac{N_{\nu_i}^{(t)} - N_{\nu_i}^n}{\Delta t} - \sum_{\mathrm{is}\in \mathrm{S}_i} \sum_{\ell} \left[\left(a^{\ell^n} N_{\nu}^{\ell^{(t)}} \right)_{\mathrm{in}} \theta_{\mathrm{is}}^{\ell^n} \right].$$

In matrix notation the molar mass equation for component ν can be expressed as

$$\sum_{\mu} \mathcal{J}_{\nu,\mu}^{(t)} \delta \mathbf{N}_{\mu}^{(t+1)} = \mathbf{b}_{\nu}^{(t)}, \quad \nu, \mu = \mathrm{w}, \mathrm{o}, \mathrm{g}. \tag{17}$$

As a simplifying step, we continue by neglecting off-diagonal blocks representing coupling between different components, i.e.

$$\frac{\partial N_{\nu}^{\ell}}{\partial N_{\mu}} = 0, \qquad \nu \neq \mu. \tag{18}$$

We get then the following decoupled systems for each component (we drop the $\cdot_{\nu,\nu}$ sub-script on the Jacobian matrix)

$$\mathcal{J}^{(t)} \delta \mathbf{N}_{\nu}^{(t+1)} = \mathbf{b}^{(t)}. \tag{19}$$

In order to solve the linear system in Eq. (19) we use the domain decomposition based iterative method presented in Sect. 3. Thus, the two-level iterated solution is now $\mathbf{N}_{\nu}^{(s)(t)}$ and the incremented solution is defined as

$$\delta \mathbf{N}_{\nu}^{(s)(t)} = \mathbf{N}_{\nu}^{(s)(t)} - \mathbf{N}_{\nu}^{(t)}. \tag{20}$$

Following Eq. (10) the updated/improved solution is then

$$\mathbf{N}_{\nu}^{(s+1/2)(t)} = \mathbf{N}_{\nu}^{(s)(t)} + \mathcal{R}^T \hat{\mathbf{d}}^{(s)}, \tag{21}$$

and the coarse grid equation for the molar mass is

$$\mathcal{R} \mathcal{J}^{(t)} \mathcal{R}^T \hat{\mathbf{d}}^{(s)} = \mathcal{R}\big(\mathbf{b}^{(t)} - \mathcal{J}^{(t)} \delta \mathbf{N}_{\nu}^{(s)(t)}\big). \tag{22}$$

The independent equations for the sub-domains $q = 1, \ldots, p$, are

$$\mathcal{J}_{qq}^{(t)} \delta \mathbf{N}_{\nu_q}^{(s+1)(t)} = \mathbf{b}_{\nu_q}^{(t)} - \sum_{r \neq q} \mathcal{J}_{qr}^{(t)} \delta \mathbf{N}_{\nu_r}^{(s+1/2)(t)}, \tag{23}$$

where summation index r includes the set of true coarse cells. The right hand side terms involving intermediate solutions $\mathbf{N}_{\nu_r}^{(s+1/2)(t)}$ are given by

$$\mathbf{N}_{\nu_c}^{(s+1)(t)} = \mathbf{N}_{\nu_c}^{(s+1/2)(t)} = \mathbf{N}_{\nu_c}^{(s)(t)} + \mathcal{R}_c^T \hat{\mathbf{d}}^{(s)}, \tag{24}$$

$$\mathbf{N}_{\nu_r}^{(s+1/2)(t)} = \mathbf{N}_{\nu_r}^{(s)(t)} + \mathcal{R}_r^T \hat{\mathbf{d}}^{(s)}, \quad r = 1, \ldots, p. \tag{25}$$

5.1 Parallel Implementation

The parallel version of the ATHENA code is based on the concept of a "simulator parallel model". The idea is that the original serial simulator is used on each sub-domain. To incorporate the solver described in Sect. 3, the main modifications consist of adding functionality for communicating between the domains. This is implemented through a Communicator class, which has the various objects to be communicated as member classes. We use the object oriented MPI library OOMPI, Squyres et al. [2003], to achieve this. OOMPI is a thin layer on top of the MPI, see e.g. Snir et al. [1996], which enables easy creation and communication of user defined objects. Since OOMPI is a fairly lightweight library and introduces little overhead, we have chosen to continue with this in the implementation of the implicit mass transport solver. In Skjellum et al. [2001] a comparison of different design strategies for an object oriented interface to MPI is given. Due to the success of MPI in defining a standard for distributed, parallel programming, the simulator is highly portable.

Following the existing framework and based on the mathematical model, we have introduced two classes inherited from OOMPI_Datatype. These are the RefinedMM class, which contains the values needed for upstream evaluation of the Jacobian matrix terms

$$a^{\ell^n}\left(\frac{\partial N_\nu^\ell}{\partial N_\nu}\right)^{(t)} \quad \text{and} \quad a^{\ell^n} N_\nu^{\ell^{(t)}}, \tag{26}$$

and the CoarseMM class, which contains the sub-domain terms contributing to the system Jacobian and right hand side given in Eq. (11).

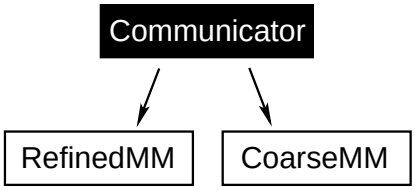

Fig. 2. The figure shows the collaboration of the Communicator class and the classes storing data between refinements, RefinedMM, and data for the coarse solve, CoarseMM.

5.2 Numerical Experiments

We perform a numerical experiment using the implicit mass transport formulation. The main goal of the experiment is to evaluate the parallel performance on various platforms. At the current stage of implementation, we map

domains onto CPUs in a one-to-one fashion. In the case where other considerations than just getting an even number of grid cells dictate the choice of domain decomposition, this approach is not optimal. Work on allowing a more flexible load balancing of the simulator is in progress.

We use SGI and IBM SP2 super computers, a dedicated IBM Linux cluster and on a network of PC workstations running Linux. Even though high-speed super computers are an important target platform for the simulator, it is also interesting to see how the code performs on lower bandwidth networks of regular workstations, as this is available to a lot of users. Consequently, we have run the simulations on three super-computing platforms and on two commodity off-the-shelf hardware platforms, see Table 1. To avoid the com-

Table 1. The table lists various hardware platforms and compilers.

index	platform	C++ compiler	MPI implementation
I	SGI Origin 3800	MIPSpro	Irix
II	IBM p690	AIX	IBM
III	IBM Linux cluster	Intel	LAM
IV	Linux cluster	GNU	MPICH
V	Linux workst. netw.	GNU	MPICH

plicating effect of calculating equilibrium in the beginning of the simulation, we do our experiments using restarts at times when equilibrium (hydrostatic) is established.

The test case domain is depicted in Fig. 3. In this case the porous medium is initially saturated with water. The simulation domain is 366m $\times$ 671m $\times$ 52m in x-, y- and z-direction respectively and is decomposed into six domains with an equal number of fine cells. Oil and gas phases migrate from one corner of the domain. For a fixed simulation time interval, we have measured the

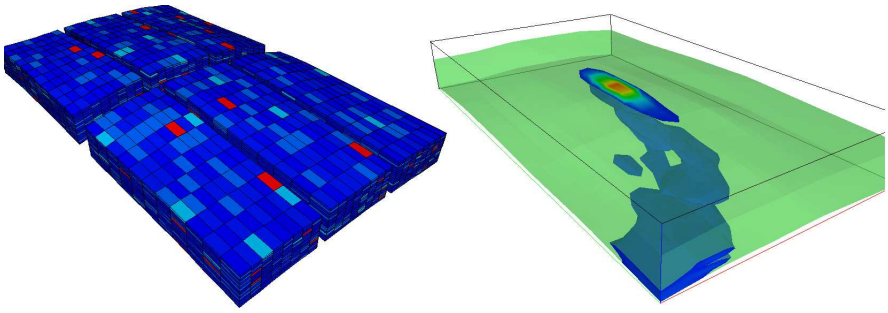

Fig. 3. The figure shows the domain decomposition (left) and gas saturation at a given time (right).

Table 2. Timing (seconds) of simulations on various platforms.

	platform I	platform II	platform III	platform IV	platform V
total	2908	764	2472	1056	1801
pressure	711	195	502	252	489
mass	2081	497	1778	725	1194

total CPU time and CPU times for *pressure solution* and *molar mass solve* separately. The results are given in Table 2. More detailed timing results, including a conceptual fault zone case, will be presented in Øian [2004].

6 A Geometrically Complex Case

We demonstrate the simulator framework in use through a multiphase flow case in a faulted porous medium, see Fig. 4. Oil and gas phases migrate through a water saturated layer and into a high permeable fault zone. The

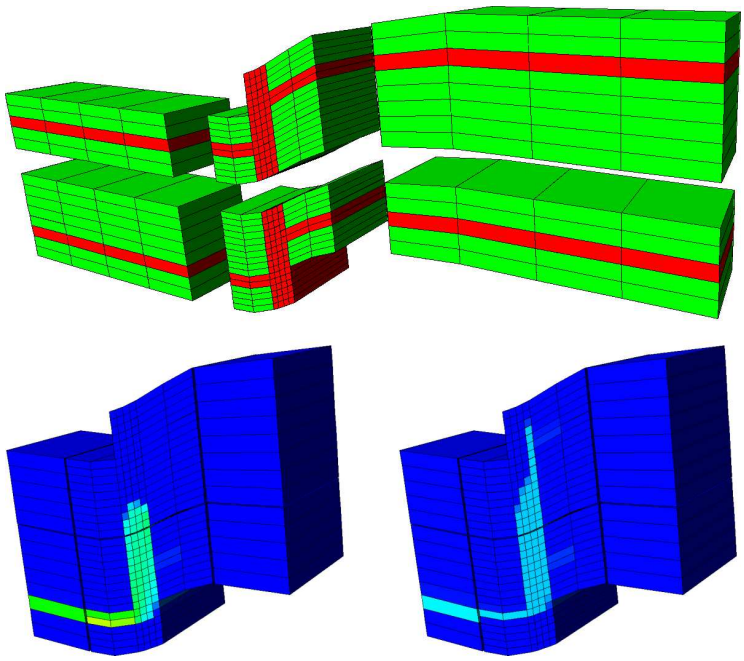

Fig. 4. The top part of the figure shows the domain decomposition including horizontal and vertical flow regions. Red color indicates high permeability. Oil and gas saturations are plotted at the same time level in the bottom left and right parts of the figure respectively, where only a middle section of the grid is visualized.

background lithology is a low permeable shale. Due to the density differences of water, oil and gas, a segregation process occurs in the vertical fault zone accompanied by much higher flow rates than in the horizontal layers.

As mentioned in Sect. 4 above, we place the transition between the horizontal and vertical flow regions inside the coarse blocks. The implicit mass transport formulation improves the allowed time step within the vertical flow region. Also, the two-level solver enables the mass transport equations to be solved in parallel, which is important if we want to make a more refined discretization. In Øian [2004] further computational results will be given.

7 Conclusion

We have presented a domain decomposition framework which is implemented in a parallel version of the flow simulator ATHENA. The two-level, iterative solver allows multiblock/multiphysics domains to be built. This is because the coarse operator is based on an algebraic combination of the fine scale operators (Galerkin) rather than an explicit coarse scale discretization. Consequently, by defining the domain decomposition a-priori, we can allow different models or discretizations in different domains. Work is in progress on applying this method on modeling flow in fractured porous media.

Another issue is that the Galerkin coarse scale operator can be viewed as an upscaled model. Traditional upscaling methods typically treat each parameter separately. Combining these into an upscaled model might be incorrect. In the multiphase case, the Galerkin operator gives a combined averaged representation by directly incorporating the fine scale, nonlinear processes caused by changes in absolute and relative permeability and capillary forces.

A sequential, implicit formulation of the mass transport equations has been implemented within the existing object oriented parallel framework. A timing experiment illustrated that the code performs better on high performance, shared memory supercomputers than distributed memory systems. This is the typical behavior for domain decomposition based methods. As we have a sequential, i.e. decoupled, solution procedure for solving the pressure and molar mass equations, the number of iterations to achieve convergence in either, depend on the time steps. The implicit molar mass equations allows larger time steps, but might introduce more iterations in the pressure solve. This will have an effect on the observed parallel efficiency since communication is involved in each iteration. Work is in progress to implement the simultaneous solution of pressure and masses, i.e. fully implicit.

A flexible load balancing scheme has not yet been implemented in the simulator. The effect of this is apparent on networks with low bandwidth connections and will also influence the scaling properties on supercomputers. With a queue system, e.g. running parallel jobs at night time, a network of workstations would still be a valuable parallel platform. Specially for development purposes and setting up simulation cases, this opportunity is useful.

Acknowledgement. The first author would like to thank Prof. L. J. Durlofsky for the opportunity to visit Stanford University, Petroleum Engineering Department, and initiating work on discrete fracture network models with Dr. Mohammad Karimi-Fard.

References

J. Aarnes, H. Reme, and M. S. Espedal. A least-squares approach for upscaling and the acceleration of a galerkin technique. Presented at the Upscaling Downunder Conference, Melbourne, Australia, 7 - 10 February, 2000.

L. Baffico, S. Bernard, Y. Maday, G. Turinici, and G. Zerah. Parallel in time molecular dynamics simulations. *Phys. Rev. E.*, 66:057701, 2002.

W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

M. Chaib, G. E. Fladmark, and M. S. Espedal. Implicit treatment of molar mass equations in secondary oil migration. *Computing and Visualization in Science*, 4(3):191–196, February 2002.

Y. Chen, L. J. Durlofsky, M. Gerritsen, and X. H. Wen. A coupled local-global upscaling approach for simulating flow in highly heterogeneous formations. *Advances in Water Resources*, (26):1041–1060, 2003.

R. Ewing, R. Lazarov, T. Lin, and Y. Lin. The mortar finite volume element methods and domain decomposition. *East-West J. Numer. Math.*, 8:93–110, 2000.

I. Garrido, M. S. Espedal, and G. E. Fladmark. A convergent algorithm for time parallelization applied to reservoir simulation. Presented at DD15, 2003.

M. Karimi-Fard, L. J. Durlofsky, and K. Aziz. An efficient discrete fracture model applicable for general purpose reservoir simulators. *SPE 79699*, 2003.

M. Karimi-Fard and A. Firoozabadi. Numerical simulation of water injection in fractured media using the discrete-fracture model and the Galerkin method. *SPE Reservoir Evaluation & Engineering*, 6(2):117–126, April 2003.

S. Knapek. Upscaling techniques based on subspace correction and coarse-grid approximation. *InSitu*, 22(1):35–58, 1998. Special issue on reservoir simulation.

S. Knapek. Matrix-dependent multigrid homogenization for diffusion problems. *SIAM J. Sci. Comp.*, 20(2):515–533, 1999.

S. H. Lee, P. Jenny, and H. A. Tchelepi. A finite-volume method with hexahedral multiblock grids for modeling flow in porous media. *Computational Geosciences*, 6:353–379, 2002.

S. H. Lee, C. Wolfsteiner, L. J. Durlofsky, P. Jenny, and H. A. Tchelepi. New developments in multiblock reservoir simulation: Black oil modeling,

nonmatching subdomains and near-well upscaling. *Society of Petroleum Engineers*, (SPE 79682), 2003.

Q. Lu, M. Peszynska, and M. F. Wheeler. A parallel multiblock black-oil model in multimodel implementation. *SPE Journal*, 7(3):278–287, September 2002. SPE 79535.

J. D. Moulton, J. E. Dendy, and J. M. Hyman. The black box multigrid numerical homogenization algorithm. *J. Comput. Phys.*, (142):80–108, 1998. Article No. CP985911.

E. Øian. *Modeling Flow in Fractured and Faulted Media*. Dr.Scient. thesis, in preparation, University of Bergen, 2004.

E. Øian, I. Garrido, M. Chaib, G. E. Fladmark, and M. S. Espedal. Modeling fractured and faulted regions: Local grid refinement methods for implicit solvers. *Computing and Visualization in Science*, 2003. Accepted for Publication.

G. Å. Øye and H. Reme. Parallelization of a compositional simulator with a galerkin coarse/fine method. In P. A. et al., editor, *Lecture Notes in Computer Science*, pages 586–594. Springer-Verlag, Berlin, 1999. LNCS 1685.

H. Reme, M. Espedal, and G. E. Fladmark. *A Preconditioning Technique as an Upscaling Procedure*, volume 131 of *The IMA Volumes in Mathematics and its Applications*, pages 283–297. Springer Verlag, Heidelberg, 2002.

H. Reme and G. Å. Øye. Use of local grid refinement and a galerkin technique to study secondary migration in fractured and faulted regions. *Computing and Visualization in Science*, 2:153–162, 1999.

H. Reme, G. Å. Øye, M. S. Espedal, and G. E. Fladmark. Parallelization of a compositional reservoir simulator. In Z.-C. S. Z. Chen, R. E. Ewing, editor, *Numerical Treatment of Multiphase Flows in Porous Media*, number 552 in Lecture Notes in Physics, pages 244–267. Springer-Verlag, Berlin, 2000.

A. Skjellum, D. G. Wooley, Z. Lu, M. Wolf, P. V. Bangalore, A. Lumsdaine, J. M. Squyres, and B. McCandless. Object-oriented analysis and design of the message passing interface. *Concurrency and Computation: Practice and Experience*, 13:245–292, 2001. (DOI: 10.1002/cpe.556).

B. F. Smith, P. E. Bjørstad, and W. Gropp. *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press, 1996.

M. Snir, S. Otto, S. Huss-Lederman, D. Walker, and J. Dongarra. *MPI: The Complete Reference*. MIT Press, 1996.

J. M. Squyres, J. Willcock, B. C. McCandless, P. W. Rijks, and A. Lumsdaine. *Object Oriented MPI (OOMPI): A C++ Class Library for MPI*. Open Systems Laboratory Pervasive Technologies Labs, Indiana University, September 3 2003. `http://www.osl.iu.edu/research/oompi/`.

R. Teigland. On some variational acceleration techniques and related methods for local refinement. *Int. J. Numer. Meth. Fluids*, (28):945–960, 1998.