Parallel aspects of a multi-domain implicit time stepping scheme for 2D miscible flow in porous media

G. Manzini\textsuperscript{a}, C. Vittoli\textsuperscript{b}, P. Wilders\textsuperscript{c}

\textsuperscript{a}CRS4, via N. Sauro 10, 09123 Cagliari, Italy
\textsuperscript{b}IBM-NAC, via Mameli 191, 09123 Cagliari, Italy
\textsuperscript{c}Delft University of Technology, PO Box 5031, 2600 GA Delft, The Netherlands

Abstract

The concentration equation in the model of miscible flow in porous media, such as appearing in reservoir engineering, is treated using unstructured finite volumes, implicit time stepping and domain decomposition. The computational effort can be demanding in particular in strongly heterogeneous reservoirs. A distributed parallel environment might be helpful here. We discuss some performance characteristics for an IBM-SP2.

1 Introduction

Miscible flow in porous media can be conveniently modeled using unstructured finite volumes for the concentration equation in combination with mixed finite elements for pressure and velocity, see Consonni et al. \cite{1}, Fotia & Manzini \cite{2} and Bergamaschi, Mantica & Saleri \cite{3}. In reservoir engineering with strongly heterogeneous reservoirs implicit time integration of the concentration equation in combination with a high resolution is a necessity, see Wilders, Fotia & Marrone \cite{4}. As a consequence, the computational effort in the concentration equation can become quite demanding and in this paper we investigate whether such a computation can be done in a distributed parallel environment. The basic mechanism is coarse-grain parallelism provided by a Schwarz-Krylov domain decomposition method.

The outline of this paper is as follows: in section 2 we briefly describe the governing equations, the discretization is discussed in section 3. We use the $\theta$-scheme in combination with a cell-centered unstructured finite-volume technique. Section 4 gives details on the domain decomposition method, i.e. Schwarz iteration accelerated with GMRES in a form taken from Brakkee & Wilders \cite{5}. In section 5 we pay
attention to some parallel aspects such as the communication scheme, implementa-
tional details and speed-up. Section 6 gives numerical experiments measuring the
speed-up. Finally, in section 7 we make some final remarks.

2 Governing Equations

The concentration equation in the model of two-dimensional single phase miscible
flow in porous media reads:

\[ \frac{\partial c}{\partial t} + \nabla \cdot F = \nabla \cdot G, \quad x \in \Omega, \quad t > 0, \]  

with \( F = [u_x c, u_y c]^T, \ G = D(u)\nabla c, \ u = [u_x, u_y]^T. \) Here, \( \Omega \) is a two-dimensional
domain representing a reservoir of constant thickness, \( c (0 \leq c \leq 1) \) is the concen-
tration of the solvent, \( \phi \) is the porosity and \( u \) is the Darcy velocity of the fluid with
Cartesian components \( u_x \) and \( u_y. \ \) \( D(u) \) is a velocity dependent dispersion tensor
given by

\[ D(u) = \frac{d_l}{|u|} \begin{pmatrix} u_x^2 & u_x u_y \\ u_x u_y & u_y^2 \end{pmatrix} + \frac{d_t}{|u|} \begin{pmatrix} u_y^2 & -u_x u_y \\ -u_x u_y & u_x^2 \end{pmatrix}, \]

with \( d_l, d_t \) coefficients measuring the longitudinal and transversal dispersivity.

Wells are modeled via appropriate boundary conditions. We employ:

\[ c = 1, \quad x \in \Gamma_{\text{in}}, \]

\[ D \nabla c \cdot n = 0, \quad x \in \Gamma \setminus \Gamma_{\text{in}}. \]

Here, \( \Gamma \) is the boundary of \( \Omega, \ n \) denotes the outward unit normal and \( \Gamma_{\text{in}} = \{ x \in \Gamma : u \cdot n < 0 \} \). To complete the model we take \( c = 0 \) as the initial condition.

The Darcy velocity is computed via the pressure equation, a second-order elliptic
equation involving the permeability \( k \) of the reservoir in its coefficients. For full
details of the model the reader is referred to Chavent & Jaffre [6]. The focus in
this paper is on the concentration equation. The sequential interaction with the
pressure equation will be neglected. This means that a stationary velocity field is
computed from the pressure equation using the mixed finite element method studied
in [3] (linear BDM element for the velocity with piecewise constant pressures). The
concentration equation is then advanced in time according to this stationary velocity
field.

3 Discretization

We employ a triangularization of \( \Omega \) with triangles \( K \); the boundary \( \partial K \) of a triangle
\( K \) consists of three edges \( e. \) The area of a triangle is denoted with \( |K| \) and the
length of the edge \( e \) by \( |e|. \) The cell-centered finite volume method leads to:

\[ \phi_K \frac{\partial c_K}{\partial t} |K| + \sum_{e \in \partial K} f_{e,K} |e| = \sum_{e \in \partial K} g_{e,K} |e|, \]
with $\phi_K$ and $c_K$ centroid values. Here, $f_{e,K}$ and $g_{e,K}$ represent the discrete approximations of the advective flux $\textbf{F} \cdot \mathbf{n}_{e,K}$ and the viscous flux $\textbf{G} \cdot \mathbf{n}_{e,K}$ on the edge $e$; ($\mathbf{n}_{e,K}$ is outward unit normal on $e$). Both $f_{e,K}$ and $g_{e,K}$ are evaluated using the edge-molecule depicted in figure 1.

Figure 1: Edge molecule, shadow volume for gradient.

Figure 2: Cell molecule for cell 0.

The advective flux is evaluated centrally with a non-linear artificial dissipation term added, similar to Jameson & Mavripilis [7]. The artificial dissipation term consists of a blending of second and fourth order differences of the unknown concentration; we apply the edge-based formulation given in Stolcis & Johnston [8] with the differentiable adaptive switch parameters taken from [4].

The viscous flux is treated similar to [4] using the shadow volume with corners the cell centers $P_1$, $P_2$ and the middle points $P_a$, $P_b$ of the lines connecting $P_4$ with $P_5$ and $P_6$ with $P_3$ (see figure 1). With this shadow volume the gradient of $c$ on the edge $e$, needed in the viscous flux, is evaluated. Values in $P_a$ and $P_b$ are just obtained by straightforward averaging of two values.

From eqn (5) it can be seen that for each triangular cell we have to sum up three edge contributions with the edge-molecule from figure 1. This leads to a cell-molecule such as sketched in figure 2. Taking eqn (5) together for all triangular volumes $K$, we end up with a system of differential equations for the centroid values of the concentration, i.e.:

$$M \frac{dc}{dt} = \psi(c, x, t),$$

with $M$ a diagonal matrix containing the cell values of the porosity $\phi$. The non-linear dependence of $\psi$ on $c$ is a consequence of the non-linearity of the artificial dissipation. Let us set $J = \frac{\partial \psi}{\partial c}$. For the time integration we consider the non-linearly implicit $\theta$-scheme in delta-formulation:

$$\left(\frac{1}{\tau}M - \theta J^n \right) \delta = \psi^n, \quad \delta = c^{n+1} - c^n.$$
222 High-Performance Computing in Engineering

Here, $\tau$ denotes the time step. We include the full Jacobian in eqn (7) and for $\theta = 1/2$ we obtain second-order accuracy in time. Equation (7) is denoted with

$$A\delta = \psi.$$  \hspace{1cm} (8)

From figure 2 it can be seen that each row of the matrix $A$ contains 10 non-zero elements (apart from boundaries). The matrix $A$ has been studied in some detail in [4] and similarly $ILU$-preconditioned $CGSTAB$ is put forward as an efficient solver.

4 Domain Decomposition

The basic mechanism providing us parallelism is a Krylov-Schwarz iteration formulated in terms of a small set of coupling equations. For every internal boundary (or interface) we define a halo consisting of those cells of which the molecule of figure 2 crosses the internal boundary. The cells in the halo are called the coupling cells and the corresponding variables are called the coupling variables. The vector of coupling variables is denoted with $\psi$. Let $Q$ denote the injection from $\psi$ into $\delta$; $Q^T$ restricts $\delta$ to $\psi$. It can be shown that block-Jacobi iteration in eqn (8) (or additive Schwarz with a minimal overlap) is equivalent with stationary Richardson iteration for the coupling equations:

$$B\psi = g, \quad B = Q^TN^{-1}AQ, \quad g = Q^TN^{-1}\psi.$$  \hspace{1cm} (9)

Here, $N$ is the block-Jacobi iteration matrix. In the case of a two-domain division $N$ and $A$ read:

$$N = \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix}, \quad A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}.$$  \hspace{1cm} (10)

The central idea is to apply GMRES to eqns (9). In mathematical sense this is equivalent with applying GMRES to eqn (8) directly with the block-Jacobi matrix defining the preconditioner. At the level of implementation the situation is slightly different; eqn (9) is less demanding with respect to operation counts and storage. In particular the latter is important for GMRES. The reader is referred to [5] and [4] for more details on the specific domain decomposition iteration.

5 Parallel aspects

The basic computation in building eqn (8) is done in a loop over the edges, see also eqn (5). Via the introduction of ghost cells we enter possible other domains. The ghost cells contain the information outside the subdomain, that are in the halo of its internal boundaries. Counting from the interface two cells in other domains are present in the halo (because we use Schwarz with a minimal overlap). On the other hand the GMRES acceleration for eqn (9) uses the vector of the coupling variables and an entry of this vector can only be attached to the subdomain it belongs to. In the present implementation, the GMRES routine runs on a host containing the global coupling vector. Having these two data-structures available, we have chosen to update the ghost cells from the latest updated version of the vector $\psi$, thus going...
through the host. This communication scheme differs from the one most commonly used, in which ghost cells are updated directly from neighbouring blocks. Our communication scheme leads to more traffic. However, the programming effort is less. It shall be clear that in our approach it is the length of the vector \( v \) that counts. Therefore, we prefer a block-wise decomposition above a strip-wise decomposition. Furthermore, in this paper blocks and processors are associated, i.e. in the parallel mode every processor contains a single subdomain. We divide both the x-interval and the y-interval in \( \sqrt{p} \) parts leading to \( p \) blocks of equal magnitude. In this paper the cases \( p = 4, 9, 16, 25 \) are treated. The code has been written in double precision Fortran 77. The parallel extension has been based upon the IBM-MPL library and communication takes place in double precision. The computational heart of the code is in the time-stepping loop. All i/o takes place on the host. In the preprocessing phase, much communication is needed. For the moment we have not included this communication in our measurements.

The speed-up \( S_p \) and the efficiency \( E_p \), measured in this paper, are defined by:

\[
S_p = \frac{\text{elapsed time on one processor}}{\text{elapsed time on } p \text{ processors}}, \quad E_p = \frac{S_p}{p}.
\]

Only the time stepping loop has been considered to determine the elapsed time. We take the same algorithm in the numerator and denominator of eqn (11). In a serial mode we are in principle able to gain speed by changing the algorithm. For example, the ddm-iteration can be taken in the Gauss-Seidel sense (multiplicative Schwartz with minimal overlap) or, if possible, we can join subdomains to end up with a single-domain computation. However, our efforts are directed towards enlarging the scale of treated problems. Large scale problems on one processor not fitting into core memory will perform badly due to extensive i/o. Therefore, even the best serial algorithm is not that visible. The main question we try to investigate is related to the scalability of the parallel algorithm during the process of enlarging both the scale of the problems and the number of processors simultaneously. Of course, our approach will not scale perfectly. The GMRES-routine is running at the host, giving some sequential computations. Furthermore, during enlarging the number of processors, the number of coupling variables will also grow faster than linear in the region \( p = 4, \ldots, 25 \). Finally, in our communication scheme, the communication time spent at the host grows linearly with the number of coupling variables.

6 Numerical experiments

In this section some numerical results for the standard test case of a quarter of five-spots are presented. Problem domain is a square of 300 meters length. An injector well and a producer well are located at the bottom left and top right corner respectively. The permeability field is based upon real field data, provided by Agip S.p.A. and shown in figure 3. The porosity was set to .1 and zero dispersion was assumed. Figure 4 sketches a typical mesh, in this case the 40-grid consisting of \( 40 \times 40 \) quadrilaterals each divided in two triangles. In Figure 5 the velocity field, such as computed with the mixed finite element method, is presented. In this paper, we take the velocity field on the 40-grid as the basic field; velocities on finer grids are derived from this field by interpolation (thus giving us the opportunity to proceed
High-Performance Computing in Engineering

without doing extensive velocity computations). Figure 5 shows a typical solution pattern shortly before breakthrough; .6 PVI corresponds with t = 60 days. All computations have been done as described in [4]. For example, θ in eqn (7) varies gradually from θ = 1 (implicit Euler) at start-up to θ = .5 (trapezoidal rule) after 20 timesteps, fixing θ afterwards. A fixed time step τ is used, with τ = .75 on the 40-grid as the basic value. For finer grids the time step is such that the Courant number is constant. In fact, the maximal Courant numbers is of the order 30 in all computations. The measurements in this paper have been done over the time interval [0, T] with T = 60 days.

![Permeability field](image1.png)  ![Computational mesh](image2.png)

Figure 3: Permeability field. Figure 4: Computational mesh.

A 30-node IBM-SP2 machine equipped with 66 MHz POWER2 processors and 128 Mbytes of memory per node was used. In order to obtain maximal communication speed over the switch we have based ourselves upon the IBM communication library (35.5 MB/sec, 40 μsec latency). Compilation was done with the restricted -O3 optimize option switched on. In table 1 we present the measurements on p processors with a constant magnitude of the subdomain grid. We have chosen for the 40-grid in the subdomains (approximately 3000 unknowns). As expected no perfect scaling is obtained. However, the final result is quite acceptable. A second interest is on using the parallel code with few processors for the present-day computations. Table 2 presents the measurements on 4 processors letting the magnitude of the subdomain problem vary. A good efficiency can be observed.

Table 1: S_p, E_p, measured speed-up and efficiency as a function of p fixing the magnitude of the subdomain.

<table>
<thead>
<tr>
<th>p=4</th>
<th>p=9</th>
<th>p=16</th>
<th>p=25</th>
</tr>
</thead>
<tbody>
<tr>
<td>S_p</td>
<td>3.4</td>
<td>5.9</td>
<td>9.3</td>
</tr>
<tr>
<td>E_p</td>
<td>.85</td>
<td>.66</td>
<td>.58</td>
</tr>
</tbody>
</table>
High-Performance Computing in Engineering 225

Figure 5: Velocity field.  
Figure 6: Concentration at .6 PVI; 40-grid, τ = .75 days.

Table 2: $S_4, E_4$, measured speed-up and efficiency on four processors as a function of the grid size in the subdomains.

<table>
<thead>
<tr>
<th></th>
<th>80-grid</th>
<th>40-grid</th>
<th>20-grid</th>
<th>10-grid</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_4$</td>
<td>3.3</td>
<td>3.4</td>
<td>3.4</td>
<td>3.2</td>
</tr>
<tr>
<td>$E_4$</td>
<td>.83</td>
<td>.84</td>
<td>.85</td>
<td>.81</td>
</tr>
</tbody>
</table>

7 Final remarks

Our results show that distributed parallel computing is useful in reservoir engineering. Present-day computations with a few subdomains can be done with nearly full speed-up. More important, we have seen the feasibility if the goal is to enlarge the scale of the treated problems. Of course, our results form only a beginning and much remains to be done. In particular, we want to study the communication strategy more closely. A next step would be to implement a parallel version of the GMRES-acceleration, enabling us to communicate between nodes directly without going through a host (at the price of some complicated coding). Although such developments will shift some boundaries, it is expected that communication will remain to play a vital role, thus forcing us to use dedicated machines with fast communication protocols and nodes having some computing power and memory.

Acknowledgments

We would like to thank AGIP S.p.A., the Italian state oil company, for providing field data. The research was carried out with the financial support of the regional authorities of Sardinia and the European Community.
References


