Parallel simulation of two-phase incompressible and immiscible flows in porous media using a
finite volume formulation and a modified IMPES approach

This content has been downloaded from IOPscience. Please scroll down to see the full text.

2010 IOP Conf. Ser.: Mater. Sci. Eng. 10 012034
(http://iopscience.iop.org/1757-899X/10/1/012034)

View the table of contents for this issue, or go to the journal homepage for more

Download details:

IP Address: 54.191.40.80
This content was downloaded on 01/09/2017 at 20:02

Please note that terms and conditions apply.

You may also be interested in:

A two-scale approach for propagating cracks in a fluid-saturated porous material
F Irzal, J J C Remmers, J M Huyghe et al.

History matching problem in reservoir engineering
Pedro González-Rodríguez, Manuel Kindelan, Miguel Moscoso et al.

A massively parallel solution strategy for efficient thermal radiation simulation
P D Nguyen, V Moureau, L Vervisch et al.

A multi-solver quasi-Newton method for the partitioned simulation of fluid-structure interaction
J Degroote, S Annerel and J Vierendeels

grim: A Flexible, Conservative Scheme for Relativistic Fluid Theories
Mani Chandra, Francois Foucart and Charles F. Gammie

Implicit large-Eddy simulation with a moving least squares-based finite volume method
Xesús Nogueira, Sofiane Khelladi, Luis Cueto-Felgueroso et al.

Incomplete Balancing Domain Decomposition for Large scale 3-D non-stationary incompressible flow
problems
Q H Yao, H Kanayama, M Ogniò et al.

Application of parallel computing to seismic damage process simulation of an arch dam
Hong Zhong, Gao Lin and Jianbo Li

Simplified TPLS as a learning tool for high-performance computational fluid dynamics
James Fannon, Jean-Christophe Loiseau, Prashant Valluri et al.
Parallel simulation of two-phase incompressible and immiscible flows in porous media using a finite volume formulation and a modified IMPES approach

R S da Silva¹, D K E de Carvalho², A R E Antunes³, P R M Lyra⁴ and R B Willmersdorf⁵

Departamento de Engenharia Mecânica, UFPE, Av. Acadêmico Hélio Ramos, s/n, CEP: 50740-530, Recife - PE – Brazil.

E-mail: ¹rogsoares@yahoo.com.br, ²dkarlo@uol.com.br, ³aantunes@yahoo.com.br, ⁴prmlyra@ufpe.br, ⁵ramiro@willmersdorf.net

Abstract. In this paper a finite volume method with a “Modified Implicit Pressure, Explicit Saturation” (MIMPES) approach is used to model the 3-D incompressible and immiscible two-phase flow of water and oil in heterogeneous and anisotropic porous media. A vertex centered finite volume method with an edge-based data structure is adopted to discretize both the elliptic pressure and the hyperbolic saturation equations using parallel computers with distributed memory. Due to the explicit solution of the saturation equation in the IMPES method, severe time step restrictions are imposed on the simulation. In order to circumvent this problem, an edge-based implementation of the MIMPES method was used. In this method, the pressure equation is solved and the velocity field is computed much less frequently than the saturation field. Following the work of Hurtado, a mean relative variation of the velocity field throughout the simulation is used to automatically control the updating process, allowing for much larger time-steps in a very simple way. In order to run large scale problems, we have developed a parallel implementation using clusters of PC’s. The simulator uses open source parallel libraries like FMDB, ParMetis and PETSc. Results of speed-up and efficiency are presented to validate the performance of the parallel simulator.

1. Introduction

One of the most popular methodologies used to describe the two-phase and sometimes the three-phase fluid flow in petroleum reservoir simulation is the IMPES (Implicit Pressure Explicit Saturation) procedure [1], [2], [6]. In this technique, a sequential time stepping procedure is used to split the computation of the pressure and the saturation fields. In the classical IMPES approach, initially, from an initial saturation distribution, the pressure equation is solved implicitly and then, the total velocity is explicitly computed from this pressure field. Following, this velocity field is used as an input for the saturation equation, which is finally solved explicitly. This process is repeated until the end of the simulation.

For the incompressible and immiscible two-phase flow of oil and water in rigid porous media, the pressure field is described by an elliptic equation that can have strong discontinuous coefficients (i.e. permeabilities) and, in general, the saturation equation is similar to a convection-diffusion type equation, in which the diffusion coefficients are associated to capillarity effects. The computation of the pressure field, at each time step, involves the solution of a system of equations which is, in general much more CPU demanding than the explicit computation of the saturation equation. On the other hand, due to explicit solution of the saturation equation, severe time restrictions are imposed on the simulation.

For large scale problems, the CPU cost of the classical IMPES procedure can become prohibitive leading researchers to find other ways to make simulation viable. In order to circumvent this problem,
we propose a parallel implementation of a modified IMPES method. In the Modified IMPES approach (MIMPES) used here, the pressure equation is solved and the velocity field is updated much less frequently than the saturation field, using the fact that, usually, the total velocity field varies slowly throughout the simulation, implying that the saturation field can be updated several times before we have to update the pressure/velocity fields [3], [8].

The governing equations are discretized by a node-centered conservative finite volume formulation with an edge-based data structure [1], [2] where geometrical coefficients, computed once in a preprocessing stage, are associated to the edges and nodes of the primal mesh to define the surface and the volume of all control volumes, respectively.

2. Mathematical model
In the present section, we briefly describe the governing equations for incompressible and immiscible, two-phase flows of water and oil through rigid porous media. This model is obtained by combining Darcy’s Law with the mass conservation equation for each phase. The model adopted here has been successfully used by many authors [1], [2], [3], [6] and [8].

Initially, we assume that the phase velocities obey the Darcy’s law, which, ignoring gravitational effects can be written for phase $i$, as

$$\vec{v}_i = -\lambda_i \nabla P_i$$

where the phase mobility tensor $\lambda_i$. Henceforth, we will assume incompressible medium and fluids. We will also ignore the capillary pressure and assume that $P = P_w = P_o$, where (w) and (o) stand, respectively, for the wetting (water) and the non-wetting (oil) phases. Additionally, conservation of mass for each phase $i$ can be written as

$$-\nabla \cdot (\rho_i \vec{v}_i) + q_i = \frac{\partial (\phi \rho_i S_i)}{\partial t}$$

In (2), $\phi$ is the porosity, i.e. fraction of the rock which can be occupied by fluids, $q_i$ denotes sources or sinks, $\rho_i$ is the phase density and $S_i$ is the saturation of phase $i$, which represents the percentage of the available pore space occupied by this phase. Due to this last definition, we can write

$$S_o + S_w = I$$

Combining (1) to (3) and after some algebraic manipulation we obtain the following pressure equation

$$\nabla \cdot (\lambda \nabla P) = -Q \text{ or } \nabla \cdot \vec{v} = Q$$

where, $\lambda = \lambda_o + \lambda_w$ is the total fluid mobility tensor, $\vec{v} = \vec{v}_o + \vec{v}_w = -\lambda \nabla P$ is the total velocity field which represent the sum of phase velocities and $Q = Q_o + Q_w$, with $Q_i = \left(q_i / \rho_i \right)$, is the total injection or production specific rate. By introducing the fractional flow function $f_i = \lambda_i / (\lambda_o + \lambda_w)$, we can also derive a hyperbolic equation for the water saturation, which can be written as
The term \( \vec{F}_y = f_y \vec{v} \) is the flux function which is dependent on the water phase saturation. As it can be seen, the pressure and saturation fields are connected through the total velocity \( \vec{v} \).

3. Numerical formulation

In order to discretize the pressure and the saturation equations, i.e., (4) and (5), respectively, we have adopted a vertex centered, median dual finite volume (FV) method, in which the coefficients necessary to our calculation are associated to the edges and to the vertex of the computational mesh [1], [2], [9]. These edge and node coefficients are pre-computed in a pre-processing stage from the more traditional element data structure which is commonly used in the finite element method.

Even though, there is, in principle, no restriction to the shape of the elements utilized to discretize the spatial domain, it is important to keep in mind that edge-based FV schemes are only linearly preserving (i.e., they exactly represent a linear field) on triangular (2D), tetrahedral (3D) or structured quadrilateral (2D) and hexahedral (3D) meshes. Therefore, extra care must be taken when using different element types, especially when considering highly distorted meshes.

The median dual control volumes adopted here are built connecting centroids of elements to middle point of the edges that surround a specific mesh node, even though different control volumes could be used (e.g. centroid dual). In edge-based vertex centered schemes, fluxes are usually integrated on the dual mesh through one or more loops over the edges, and the computational cost is, essentially, proportional to the number of edges of the mesh. To properly handle porous media discontinuities, we perform the integration over the whole domain in a sub-domain by sub-domain approach, where a sub-domain is defined by a group of elements that share the same physical properties such as permeability and porosity.

In this work, we have chosen to use the cell distributed methodology due to the easiness of associating rock properties to sub-domains which naturally fit to reservoir bed boundaries. For a detailed description of all steps to obtain the discretized pressure, velocity and saturation equations see [1] and [2].

3.1. Implicit pressure equation

In (6), we present the discretized form of the pressure equation for 2-D and 3-D problems.

\[
\sum_{k=1}^{N_{\text{dom}}} \left[ \sum_{i=1}^{N_{\text{NN}}(\Omega_R)} \left( -\frac{1}{2} \left( \nabla \hat{p}_{ij} \cdot \nabla \hat{p}_{ij} + \nabla \hat{p}_{ij} \cdot \nabla \hat{p}_{ij} \right) - \frac{\left( \nabla \hat{p}_{ij} \cdot \nabla \hat{p}_{ij} \right)}{2} \left( \bar{E}_{ij} + \frac{1}{\Delta_{ij}} \left( \bar{P}_{ij} - \bar{P}_j \right) \right) \right] \cdot \bar{v}_{ij} \right] = \sum_{i=1}^{N_{\text{dom}}} Q_i \cdot V_i
\]  

The terms of equations \( N_{\text{dom}}, \nabla \hat{p}_j, \bar{E}_{ij}, \bar{P}_{ij}, Q_i, V_i \) are, respectively, number of domains, number of neighbors nodes connected to node \( I \), the domain, approximated pressure gradient, \( \bar{E}_{ij} \) edge vector, orthogonal vector to control surface, source/sink term and nodal control volume.

In this work, all calculations are performed using a parallel reservoir simulator written in C++ which we have incorporated some open source packages, such as the FMDB, used for mesh management, the ParMetis, used for mesh partitioning and the PETSc used as a library of linear solvers and pre-conditioners. Due the fact that the discretization of the pressure equation generates a sparse and non-symmetric matrix, we have used the Generalized Minimum Residual Solver (GMRES)
and the ASM (Additive Schwarz Method) as pre-conditioner. PETSc solver default parameters such as convergence tolerance \((10^{-8})\) and GMRES restart number \((30)\) were kept. After the pressure field is computed, the mid-edge velocity field is obtained by (7).

\[
\hat{v}^{\Omega}_{\mu L} = -\bar{\lambda}^{\Omega}_{\mu L} \left( \frac{\nabla P^{\Omega}_{I} + \nabla P^{\Omega}_{J}}{2} \right) - \left( \frac{\nabla P^{\Omega}_{I} + \nabla P^{\Omega}_{J}}{2} \right) \cdot \bar{L}^{\mu L} + \left( \frac{P_{I} - P_{I}}{\Delta^{\mu L}} \right) \bar{L}^{\mu L}
\]  

(7)

In this work, we have dealt only with isotropic porous media, even though there is no restriction to handle anisotropic media, therefore we have assumed that \(\bar{\lambda}^{\Omega}_{\mu L} = K^{\Omega}_{\mu L} \bar{\lambda}_{\mu L}, \) where \(K^{\Omega}_{\mu L} = k^{\Omega}_{L} I\), with \(k^{\Omega}_{L}\) = constant for each sub-domain \(\Omega_{R}\), and \(I\) is the identity matrix. The edge values of the scalar mobility terms are approximated using a mid-point rule in order to formally guarantee second order accuracy, i.e. \(\bar{\lambda}_{\mu L} = \left( \bar{\lambda} + \lambda_{I} \right) / 2\) and viscosity is constant under the assumption of incompressible and isothermal flow. And now, we can redefine (6) using flux function approximation as

\[
\sum_{R=1}^{N_{DOM}} \left( \sum_{t_{J} \in \Omega_{J}} \tilde{F}^{\Omega}_{\mu L} \cdot \tilde{C}^{\Omega}_{\mu L} + \sum_{t_{J} \in \Omega_{J}} \tilde{F}_{\mu L}^{\mu L} \cdot \tilde{D}^{\Omega}_{\mu L} \right) = \sum_{R=1}^{N_{DOM}} Q^{\Omega}_{I} V^{\Omega}_{I}
\]  

(8)

3.2. Explicit saturation equation

Usually, in petroleum reservoir simulators, the discretization of the advective term that characterize the hyperbolic saturation equation is performed by the classical first order upwind type method, which is capable of completely eliminating spurious oscillations at the cost of introducing a large amount of artificial diffusion [6]. On the other hand, pure second order schemes produce physically unrealistic results, with overshoots and/or undershoot in the vicinity of sudden changes in the saturation field (i.e. shocks). By integrating (5) and applying the divergence theorem we can write

\[
\int_{\Omega} \phi \frac{\partial S}{\partial t} d\Omega + \int_{\Gamma} \tilde{F}_{w} \left( S_{w} \right) \cdot \tilde{n} d\Gamma = \int_{\Omega} Q_{w} d\Omega
\]  

(9)

which its discretized form is given by (10).

\[
\hat{S}_{w}^{n+1} = \hat{S}_{w}^{n} - \sum_{r=1}^{N_{DOM}} \Delta t \phi^{\Omega_{I}} V^{\Omega_{I}} \left( \sum_{L=1}^{N^{\Omega_{I}}} \tilde{F}_{\mu L}(w) \cdot \tilde{C}_{\mu L} + \sum_{L=1}^{N^{\mu L}} \tilde{F}_{\mu L}(w) \cdot \tilde{D}_{\mu L} + Q^{\Omega}_{I} V^{\Omega}_{I} \right)
\]  

(10)

The source term, which was treated using a simple fractional step approach, is non zero only at production wells and for a particular mesh node \(I\), the second term in the left hand side is approximated as:

\[
\int_{\Gamma_{I}} \tilde{F}_{w} \left( S_{w} \right) \cdot \tilde{n} d\Gamma = \sum_{L_{I}} \frac{1}{2} \left[ \tilde{F}_{I}^{+} \left( S_{+}^{r} \right) + \tilde{F}_{I}^{-} \left( S_{-}^{r} \right) \right] \cdot \tilde{C}_{\mu L} - \alpha_{\mu L} \left( S_{+}^{r} - S_{-}^{r} \right)
\]  

(11)
where $\alpha_{IJ} = \left| \hat{L}_{IJ} \right| \left| \Delta f_{IJ(w)} / \Delta S_{IJ(w)} \right|$, with $\Delta f_{IJ(w)} / \Delta S_{IJ(w)} = \left( f_{IJ(w)} - f_{I(w)} \right) / \left( S_{IJ(w)} - S_{I(w)} \right)$ and the superscripts (-) and (+) are used to indicate that fluxes are computed using the following linear extrapolated saturation values (12).

$$S_{I(w)}^+ = S_{I(w)} + \frac{\psi_L^+}{2} \left( \nabla S_{I(w)} \cdot \overrightarrow{L} \right)$$ and $$S_{I(w)}^- = S_{I(w)} + \frac{\psi_L^-}{2} \left( \nabla S_{I(w)} \cdot \overrightarrow{L} \right)$$ (12)

Where $\overrightarrow{L}_{IJ}$ is the length vector in the edge direction (i.e. $\vec{x}_{IJ} - \vec{x}_I$), and $\psi_L$ is a slope limiter which must smoothly switch from one (second order scheme) to zero (first order scheme) in the vicinity of saturation shocks. $\psi_L$ is responsible for switching the scheme from second order to first order whenever necessary and $\psi_{IJ}$ is responsible for the edge interpolative boundedness, i.e., it guarantees that the extrapolated values of the saturation values throughout the edge remain between $S_{I(w)}$ and $S_{IJ(w)}$.

As previously mentioned, we have chosen this method due to its robustness and relatively low computational cost. Whenever using elements with high aspect ratios, which are common in mesh adaptive processes, other alternatives, such as the gradient extrapolation approach or the artificial dissipation scheme have, respectively, produced erroneous solutions with noticeable over and undershoots or overly diffusive solutions [1], [2].

4. Modified IMPES approach

The IMPES method is a segregated type method in which the flow equations are manipulated in order to produce an elliptic pressure equation, solved implicitly and a hyperbolic type saturation equation, which is then solved explicitly. In classical IMPES method the pressure and the saturation fields are updated assuming a single time step.

On the other hand, the Modified Implicit Pressure Explicit Saturation approach (MIMPES) consists in assigning larger time steps ($\Delta t_p$) to the implicit pressure equation than those used to solve the saturation equation ($\Delta t_{CFL}$) which is constrained by CFL condition. The latter is solved repeatedly until the saturation filed reaches the same instant of the pressure field, i.e. the summation of all $\Delta t_{CFL}$ must be equal to $\Delta t_p$. Then, a new $\Delta t_p$ is calculated and a new summation of $\Delta t_{CFL}$ is performed. The major advantage of this strategy is that the implicit pressure equation, which represents more than 90% of all calculations, is solved several times less than the explicit saturation equation.

The MIMPES algorithm, we are using in the present paper, is an edge-based implementation of the original element-based algorithm proposed by [8] with a slightly modification. The time step control strategy is based on the velocity field variation. An algorithm would look like the following:

1. Calculate pressure field;
2. $\Delta t_{sum} = 0$
3. Calculate velocity field;
4. Calculate $\Delta t_{CFL}$;
5. $\Delta t_{sum} = \Delta t_{sum} + \Delta t_{CFL}$;
6. $\Delta t^{n+1}_p = \frac{DVTOL}{\left| \Delta \vec{V} \right|} \Delta t^n_p$;
7. if $\Delta t_{\text{sum}} \geq \Delta t_{p}^{n+1}$
   go to step 1
else
   go to step 8
8. Calculate saturation field
9. Go to step 3.

After calculating the velocity field, a $L_2$ norm, $\|\Delta \vec{v}_T\|$, of all mid-edge velocities is calculated. The new time-step ($\Delta t_{p}^{n+1}$) for the pressure equation is computed through

$$\Delta t_{p}^{n+1} = \frac{DVTOL}{\|\Delta \vec{v}_T\|} \Delta t_{p}^{n} \quad (13)$$

where $DVTOL$ represents an empiric number that turns simulation more conservative (closer to classical IMPES and consequently slower) or more dynamic (faster but less accurate). Previous tests with different DVTOL values [1] showed that 0.05 can give good results, i.e., faster than classical IMPES and with acceptable accuracy.

At the very first simulation time step, $\Delta t_{p}^{n}$ assumes the value defined by the CFL restriction of the explicit formulation adopted to solve the saturation equation, i.e., $\Delta t_{p}^{n=0} = \Delta t_{CFL}$.

To avoid too large time-steps for the pressure field, which could affect the accuracy, or too small time-steps that would diminish the efficiency of the procedure, the following control procedure [8] was adopted.

$$0.75 \leq \frac{\Delta t_{p}^{n+1}}{\Delta t_{p}^{n}} \leq 1.25 \quad (14)$$

Therefore, if $\Delta t_{p}^{n+1}$ is calculated by (13), it is out of the bounds determined by (14), the new implicit time-step is calculated as follows: $\Delta t_{p}^{n+1} = 0.75 \Delta t_{p}^{n}$ or $\Delta t_{p}^{n+1} = 1.25 \Delta t_{p}^{n}$. In order to use this methodology in a safety way, it is strongly recommended to check its effect on the accuracy of the numerical solutions in order to obtain the best accuracy/CPU time ratio.

Although the pressure equation is not function of time unlike saturation, it must be emphasize here that $\Delta t_{p}^{n}$ represents how long pressure field will be held constant while saturation field is calculated for successive time steps.

5. Parallel implementation issues

5.1. Partition and load balance

Parallel simulations with unstructured meshes need a special attention to distribute nodes and elements among processors. The first step towards parallel efficiency is guarantee that each partition receives the same number of nodes to avoid load unbalance which represents a case where a processor works much more than others. The second step concerns about the interface among processors where the number of nodes with remote copies must be minimum to reduce the overhead caused by the parallel communications. Figure 1 shows an example of a partitioned mesh without and with load balance.
Figure 1. Example of a 3-D mesh of tetrahedral distributed among five processors. Without (a) and with (b) load balance performed through ParMetis.

5.2. Data structure for distributed meshes
The use of unstructured meshes by parallel simulators demands a mesh manager able to perform some tasks like data migration among processors to satisfy the load balance.

The open source library FMDB (Flexible Distributed Mesh Database) is a parallel mesh manager written in C++ which uses the libraries ParMetis and Autopack for mesh partitioning and efficiently parallel message passing, respectively.

5.3. Parallel iterative solvers and pre-conditioners
Numerical simulations can lead to millions or even billions of degrees of freedom which are commonly solved by algebraic system of equations. The PETSc (Portable, Extensible, Toolkit for Scientific Computation) library provides a set of parallel/sequential iterative solvers and pre-conditioners which can lead with large communication sparse matrices.

In the present work, the diffusion term given by (7) can be written in the following matrix form:

\[
EFx + Gx = Ax = RHS
\]  

(15)

where \([G]\) is the parallel flux projection to the edge direction and the product \([E][F]\) is the orthogonal flux projection to the edge.

Based on the fact that solvers like GMRES or Conjugated Gradient performs their calculations using only matrix-vector products, PETSc provides a matrix-free scheme to solve a system of equation without assembling explicitly a matrix of coefficients. Three matrix-vector products and one vector-vector addition are required to produce the final \(Ax\) matrix-vector product. A function pointer, which will return the matrix-vector product, must be passed to PETSc through MatShellSetOperation so every time a Krylov sub-space based solver need to computer the product \(Ax\), the user provided function will be called.

6. Numerical results
This problem, which was adapted from [5] consists in a ¼ of five spot problem. The porous media is assumed to be homogeneous and isotropic with \(K = I\) throughout the whole domain. We also assume that porosity is homogeneous even though its actual value is not relevant because we are only using it to define the non-dimensional time or PVI (Pore Volumes Injected) which is given by (16) and where \(V_p\) and \(Q\) are the total porous volume and the total flow rate, respectively.
\[ PVI = \int \frac{Q \, dt}{V_p} \]  

Water and oil viscosities are, respectively, \( \mu_w = 1.0 \) and \( \mu_o = 4.0 \), therefore, the viscosity ratio (essentially the mobility ratio) is \( M = \left( \frac{\mu_o}{\mu_w} \right) = 4.0 \). Boundary conditions are no-flux at the external boundaries, \( S_j = 1.0 \) in the injection well, and \( P_w = P_{br} = 0.0 \) at the upper left and bottom right corners. A computational mesh with 45,459 nodes and 232,084 tetrahedral elements has been used to simulate the two phase oil-water flow for 1.0 PVI.

Durlhofsky [5] solved this problem using a combined mixed finite element (used to solve the pressure/velocity problem) and finite volume (used to solve the saturation problem) approach. Figure 3 presents the saturation field for three moments of simulation in parallel using 8 processors and the MIMPES approach with DVTOL = 0.05: \( t = 0.1 \) PVI figure (3a), \( t = 0.45 \) PVI (water-cutting) figure 3b and \( t = 0.6 \) PVI figure 3c. All figures show mesh partitions separated purposely to highlight each one.

**Figure 3.** Saturation fields at different simulation times: 0.1 PVI (a), 0.45 PVI (b) and 0.6 PVI (c). These results were obtained using the MIMPES approach for 8 processors. Mesh partitions were separated purposely to highlight each one.

![Saturation fields](image)

**Figure 3.** Oil productivity: recovered (a) and accumulated (b) oil analysis highlighting the water cutting moment.

Figure 3a and 3b show a comparison between both approaches related to oil productivity: recovered oil and accumulated oil, respectively. The x-axis range from both figures varies from 0.4 to
1.0 PVI to highlight the accuracy of MIMPES compared to the IMPES approach during the water-cutting. The recovered oil plot shows the relative oil flow through production well. In the beginning, the fluid flow production is 100% oil until the breakthrough (the moment when the water phase reaches the production well). The second plot, figure 3b, shows the relative oil production related to the original oil in place, i.e., the quantity of oil exploited from the reservoir.

In figure 4a, we compared the IMPES and MIMPES approaches sequentially (one core) to show how fast the later is compared to the former. Then, the MIMPES simulation is repeated using different numbers of cores (2, 4, 8, 16 and 32) to obtain the speed-up curve showed in figure 4b.

The speed-up curve 4b tells us how fast a parallel simulation is when compared to the sequential running. In general, parallel communication overhead among processes increases as more cores are used which leads the plot to depart from the ideal curve.

**Figure 4.** Comparison between simulations approaches (a): IMPES and MIMPES. The former is performed only sequentially and the later with 1, 2, 4, 8, 16 and 32 processors. Speed-up curve (b) showing performance of the parallel simulation with the MIMPES approach with 1, 2, 4, 8, 16 and 32 processors.

In Figure 4a total simulation time (in hours) of both approaches are compared. In the blue column, the classical IMPES, and, in the red columns, the MIMPES. With 32 cores, the MIMPES was 44 times faster than the IMPES with one core which took more than one day to complete the simulation while the former finished the same simulation in less than an hour.

7. Conclusions

In the present paper, we have briefly presented a node-centered, edge-based, higher order finite volume method with a “Modified Implicit Pressure, Explicit Saturation” (MIMPES) using parallel computers with distributed memory capable to model the 3-D incompressible and immiscible two-phase flow of water and oil porous media. This approach produced very acceptable results with a considerable reduction in CPU time. The use of parallel computers turned this approach even more attractive.

Acknowledgments

The authors would like to thank the following agencies: “Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES)”, “Agência Nacional de Petróleo (ANP PRH-26)”, the “Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), to “Núcleo de Atendimento em
Computação de Alto Desempenho” (NACAD) COPPE/RJ, to “Centro Nacional de Processamento de Alto Desempenho em São Paulo CENAPAD-Campinas” and to “Fundação de Amparo à Ciência e Tecnologia do Estado de Pernambuco (FACEPE)”, for their financial and infrastructure support during the development of this work.

8. References


