A Fully Implicit, Compositional, Parallel Simulator for IOR Processes in Fractured Reservoirs

Reza Naimi-Tajdar,[*] SPE, **Choongyong Han,** SPE, **Kamy Sepehrnoori,** SPE, **Todd J. Arbogast,** SPE,
and **Mark A. Miller,**[**] SPE, the University of Texas at Austin

Summary

Naturally fractured reservoirs contain a significant amount of the world oil reserves. A number of these reservoirs contain several billion barrels of oil. Accurate and efficient reservoir simulation of naturally fractured reservoirs is one of the most important, challenging, and computationally intensive problems in reservoir engineering. Parallel reservoir simulators developed for naturally fractured reservoirs can effectively address the computational problem.

A new accurate parallel simulator for large-scale naturally fractured reservoirs, capable of modeling fluid flow in both rock matrix and fractures, has been developed. The simulator is a parallel, 3D, fully implicit, equation-of-state compositional model that solves very large, sparse linear systems arising from discretization of the governing partial differential equations. A generalized dual-porosity model, the multiple-interacting-continua (MINC), has been implemented in this simulator. The matrix blocks are discretized into subgrids in both horizontal and vertical directions to offer a more accurate transient flow description in matrix blocks. We believe this implementation has led to a unique and powerful reservoir simulator that can be used by small and large oil producers to help them in the design and prediction of complex gas and waterflooding processes on their desktops or a cluster of computers. Some features of this simulator, such as modeling both gas and water processes and the ability of 2D matrix subgridding to the best of our knowledge are not available in any commercial simulator. The code was developed on a cluster of processors, which has proven to be a very efficient and convenient resource for developing parallel programs.

The results were successfully verified against analytical solutions and commercial simulators (ECLIPSE and GEM). Excellent results were achieved for a variety of reservoir case studies. Applications of this model for several IOR processes (including gas and water injection) are demonstrated. Results using the simulator on a cluster of processors are also presented. Excellent speedup ratios were obtained.

Introduction

The dual-porosity model is one of the most widely used conceptual models for simulating naturally fractured reservoirs. In the dual-porosity model, two types of porosity are present in a rock volume: fracture and matrix. Matrix blocks are surrounded by fractures and the system is visualized as a set of stacked volumes, representing matrix blocks separated by fractures **(Fig. 1).** There is no communication between matrix blocks in this model, and the fracture network is continuous. Matrix blocks do communicate with the fractures that surround them. A mass balance for each of the media yields two continuity equations that are connected by matrix-fracture transfer functions which characterize fluid flow between matrix blocks and fractures. The performance of dual-porosity simulators is largely determined by the accuracy of this transfer function.

The dual-porosity continuum approach was first proposed by Barenblatt et al. (1960) for a single-phase system. Later, Warren and Root (1963) used this approach to develop a pressure-transient analysis method for naturally fractured reservoirs. Kazemi *et al.* (1976) extended the Warren and Root method to multiphase flow using a 2D, two-phase, black-oil formulation. The two equations were then linked by means of a matrix-fracture transfer function. Since the publication of Kazemi et al. (1976), the dual-porosity

[*] Now with BP.
[**] Now with Lucid Reservoir Technologies.

approach has been widely used in the industry to develop field-scale reservoir simulation models for naturally fractured reservoir performance (Thomas et al. 1983; Gilman and Kazemi 1983; Dean and Lo 1988; Beckner et al. 1988; Rossen and Shen 1989).

In simulating a fractured reservoir, we are faced with the fact that matrix blocks may contain well over 90% of the total oil reserve. The primary problem of oil recovery from a fractured reservoir is essentially that of extracting oil from these matrix blocks. Therefore it is crucial to understand the mechanisms that take place in matrix blocks and to simulate these processes within their container as accurately as possible. Discretizing the matrix blocks into subgrids or subdomains is a very good solution to accurately take into account transient and spatially nonlinear flow behavior in the matrix blocks. The resulting finite-difference equations are solved along with the fracture equations to calculate matrix-fracture transfer flow. The way that matrix blocks are discretized varies in the proposed models, but the objective is to accurately model pressure and saturation gradients in the matrix blocks (Saidi 1975; Gilman and Kazemi 1983; Gilman 1986; Pruess and Narasimhan 1985; Wu and Pruess 1988; Chen et al. 1987; Douglas et al. 1989; Beckner et al. 1991; Aldejain 1999).

As a generalization of the dual-porosity concept, Pruess and Narasimhan (1985) developed a “Multiple Interacting Continua” method (MINC), which treats the multiphase and multidimensional transient flow in both fractures and matrix blocks by a numerical approach. The transient interaction between matrix and fracture is treated in a realistic way. The main assumption in the MINC method is that thermodynamic conditions in the matrix depend primarily on the distance from the nearest fracture (Pruess and Narasimhan 1985). Therefore, one can partition the flow domain into compositional volume elements in such a way that all interfaces between volume elements in the matrix are parallel to the nearest fracture. Subgridding of matrix blocks on the basis of distance from the fractures gives rise to a pattern of nested volume elements. For the 2D case, it is shown in **Fig. 2.** Each volume element has a defined thermodynamic state assigned to it. The basic MINC concept of partitioning matrix blocks according to distance from the fracture faces can be extended readily to arbitrary irregular block shapes and sizes. Limitations of dual-porosity model and MINC assumptions are well known and documented. Some of these issues such as capillary continuity and fluid flow between direct matrix-to-matrix connections could be answered by dual-permeability or discrete fracture network models, which are outside of the scope of this study.

Subgridding dramatically increases computer time and storage, especially when all the flow equations are solved implicitly. While serial computing and simulation technology may be adequate for typical reservoirs, naturally fractured reservoirs need more gridblocks to adequately define the transient flow in matrix media. Parallel reservoir simulation technology places a powerful tool in the hands of reservoir engineers and geologists for determining accurate fluid in-place, sweep, and reservoir performance. MINC method has been used in other black oil and compositional simulators (such as ECLIPSE and STARS) with parallel solvers. CMG’s STARS thermal model uses compositionally-dependent K-values and works on multithreaded, shared memory computers.

The primary objectives of this study are (1) to develop a new, parallel, equation-of-state compositional, fully implicit simulator to model three-phase fluid flow in naturally fractured oil reservoirs for IOR processes, (2) to verify and test the developed model against analytical solutions and commercial simulators, and (3) to investigate the parallel computational efficiency of the developed model.

Mathematical Formulation

In the dual-porosity model, two overlapping continua, one corresponding to the fracture medium and one corresponding to the matrix medium, are considered. Thus two values of most variables and parameters are attributed to each spatial location. The equations of motion and of component mole conservation are written independently for each medium and should hold at every point of the fracture and matrix medium and at all times. The four most important transport mechanisms occurring in permeable media are viscous forces, gravity forces, dispersion (diffusion), and capillary forces (Lake 1989). Transfer of fluids between the two media is taken into consideration by a source/sink (transfer) function.

Isothermal multicomponent and multiphase flow in a porous medium can be described using three different types of equations: component conservation equations, equations constraining volume and component moles, and phase-equilibrium equations dealing with equilibrium component mass transfer between phases, in which flash calculations using an EOS are performed to determine amounts and compositions of equilibrium phases.

Neglecting dispersion and mutual solubility between water and hydrocarbon phases, for a system consisting of n_c hydrocarbon components and n_p fluid phases (excluding the aqueous phase), these three types of equations are mathematically expressed for a control volume in the following sections.

Component Mole Conservation Equations. Darcy's law is a fundamental relationship describing the flow of fluids in permeable media. The differential form of Darcy's law can be used to treat multiphase unsteady-state flow, nonuniform permeability, and nonuniform pressure gradients. It is used to govern the transport of phases from one cell to another under the local pressure gradient, rock permeability, relative permeability, and viscosity. In terms of moles per unit time, the hydrocarbon component conservation equations, for both the fracture and matrix systems, are the following:

Fracture system (subscript *f*):

$$V_b \frac{\partial}{\partial t}\left(\phi_f N_{f\,i}\right) - V_b \vec{\nabla}\cdot\sum_{j=1}^{n_p}\lambda_{fj}\xi_{fj}x_{f\,ij}\left(\vec{\nabla}P_{fj} - \gamma_{fj}\vec{\nabla}D_f\right) - q_i + \tau_{mf\,i} = 0 \quad \ldots\ldots(1)$$

Matrix system (subscript *m*):

$$V_b \frac{\partial}{\partial t}\left(\phi_m N_{mi}\right) - V_b \vec{\nabla}\cdot\sum_{j=1}^{n_p}\lambda_{mj}\xi_{mj}x_{mij}\left(\vec{\nabla}P_{mj} - \gamma_{mj}\vec{\nabla}D_m\right) = 0 \quad \ldots(2)$$

For $i = 1,2,\ldots,n_c$

These equations also hold for water by inserting the properties of the aqueous phase.

Volume Constraint Equations. The volume constraint states that the pore volume in each cell must be filled completely by the total fluid volume (including aqueous phase). The volume constraint equations for both fracture and matrix media are the same and are as follows:

$$\sum_{i=1}^{n_c+1} N_{f\,i}\sum_{j=1}^{n_p+1} L_j\bar{v}_j = 1 \text{ (fracture system) } \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3)$$

$$\sum_{i=1}^{n_c+1} N_{mi}\sum_{j=1}^{n_p+1} L_j\bar{v}_j = 1 \text{ (matrix system). } \ldots\ldots\ldots\ldots\ldots\ldots\ldots..(4)$$

Phase-Equilibrium Equations. The equilibrium solution must satisfy three conditions (Baker et al. 1982). First, the molar balance constraint must be preserved. Second, the chemical potentials for each component must be the same in all phases. Third, the Gibbs free energy at constant temperature and pressure must be a minimum. With the assumption of local thermodynamic equilibrium for the hydrocarbon phases, the criterion of phase equilibrium applies:

$$\ln\left(f_{f\,i}^{g}\right) - \ln\left(f_{f\,i}^{o}\right) = 0 \text{ (fracture system). } \ldots\ldots\ldots\ldots\ldots\ldots..(5)$$

$$\ln\left(f_{mi}^{g}\right) - \ln\left(f_{mi}^{o}\right) = 0 \text{ (matrix system). } \ldots\ldots\ldots\ldots\ldots\ldots\ldots(6)$$

Independent Variables. Eqs. 1 through 6 describe the fluid flow through porous media in naturally fractured reservoirs. There are 2(2n_c+2) equations. Independent unknowns are chosen as lnK_i, N_i, P_w, N_w (N = moles per unit pore volume) in each medium, fracture and matrix, which gives 2(2n_c+2) primary variables. This set of independent variables is likely to be the best choice largely because it makes the fugacity equations nearly linear (Wang et al. 1999). All the fluid-related properties and variables in Eqs. 1 through 6 can be expressed as a function of the selected independent variables.

Transfer Function and Boundary Conditions. The transfer function for each component (hydrocarbon components and water), $\tau_{mf\,i}$ are evaluated at the boundary between the matrix and fracture media and have the following forms:

$$\tau_{mf\,i} = NM \cdot V_b \sum_{l=1}^{N_b} \frac{\partial}{\partial t}\left(\phi_m N_{mi}\right)_l \,, \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(7)$$

where *NM* is the number of matrix blocks within a fracture gridblock (may be a fractional number), and N_b is the number of matrix subgrids. No-flow boundary conditions for component mole conservation equations in the fracture system are considered. The boundary condition for matrix blocks is continuity of all phase pressures.

Solution Approach

The material balance equations (Eqs. 1 and 2) need to be discretized using a proper scheme for a given grid system that represents the geometry of the reservoir. A fully implicit solution method is used to solve the governing equations. This method treats each term in Eqs. 1 and 2 implicitly. These equations are nonlinear and must be solved iteratively. A Newton procedure is used, in which the system of nonlinear equations is approximated by a system of linear equations. An analytical method is used to calculate the elements of the Jacobian matrix (a matrix that forms the system of linear equations and whose elements are the derivatives of the governing equations with respect to the independent variables). Using the Schur complement method (described in the next section), the matrix equations in each fracture cell are condensed and added to the diagonal elements of the fracture system. Finally, the nonlinear fracture equations are solved by one of the linear solvers of PETSc (Portable Extensible Toolkit for Scientific Computation). To solve the governing equations for the independent variables over a timestep, we take the following steps.

1. Initialization of fluid pressure and total compositions in both fracture and matrix cells.
2. Determination of phase properties and phase state. Flash calculations are performed for each cell to determine phase compositions and densities. The states of all phases present are determined as gas, oil, or aqueous. Phase viscosities and relative permeabilities are then computed.
3. Linearization of the governing equations for both fracture and matrix media. All the governing equations are linearized in terms of the independent variables, and the Jacobian matrices for both fracture and matrix media are formed. The elements of the Jacobian are computed.
4. Decomposition of the matrix medium from the fracture medium and reduction of the linear system. The Schur complement method is used to decouple the matrix system from the fracture system (for more details, see the next section). This reduces the linear system to only the fracture system.
5. Solution of the reduced system of the linear equations (fracture system) for the independent variables in the fracture medium.
6. Solution of the decoupled equations for the independent variables in the matrix medium.
7. Updating the physical properties.
8. Checking for convergence. The residuals of the linear system obtained in Step 3, which contain both fracture and matrix media, are used to determine convergence. If a tolerance is exceeded, the elements of the Jacobian and the residuals of the governing equations are then updated and another Newton iteration is performed by returning to Step

4. If the tolerance is met, a new timestep is then started by returning to Step 3.

Another criterion for convergence can be used based on the pressure and saturation changes (ΔP_w and ΔS_j) being "sufficiently small" for a given change of independent variables by Newton iteration. This criterion is preferred especially when large saturation changes may occur for very small changes in composition during Newton iteration.

One of technical challenges of this solution approach is to derive manually derivatives of all terms in the governing equations with respect to independent variables in order to construct a Jacobian matrix. The chain rule is extensively used for the derivation. For example, derivatives of phase saturation are evaluated using derivatives of total moles of the phase per pore volume and molar density of the phase.

Schur Complement Method. Let the number of fracture unknowns be *I,* and denote them by *F.* Associated with each gridblock *i=1,2,...,I,* we have a series of matrix unknowns *M.* After linearization by the Newton method, fracture and matrix equations can be summarized by

$$A_{ff}F + B_{fm}M = B_f \quad \text{(fracture system).} \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (8)$$

$$C_{mf}F + D_{mm}M = B_m \quad \text{(matrix system).} \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (9)$$

By solving for *M* in the matrix equations (Eq. 9), we obtain

$$M = D_{mm}^{-1}\left(B_m - C_{mf}F\right). \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (10)$$

By substitution of *M* (Eq. 10) into the fracture equations (Eq. 8), we obtain

$$A_{ff}F + B_{fm}D_{mm}^{-1}\left(B_m - C_{mf}F\right) = B_f. \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots (11)$$

And by solving Eq. 11 for *F* we obtain

$$\left(A_{ff} - B_{fm}D_{mm}^{-1}C_{mf}\right)F = B_f - B_{fm}D_{mm}^{-1}B_m. \ \ldots\ldots\ldots\ldots\ldots (12)$$

In Eq. 12, there are no matrix unknowns (*M*). They have been reduced to a system of fracture equations only (i.e., single-porosity model). In Eq. 12, the challenge is to calculate terms ($B_{fm}D^{-1}{}_{mm}C_m$) and ($B_{fm}D^{-1}{}_{mm}B_m$). The most time- and memory-consuming part of the calculations is the solution of the inverse of the Jacobian matrix for the matrix media $D^{-1}{}_{mm}$. By considering Eqs. 10 and 12, we actually need $D^{-1}{}_{mm}C_{mf}$ and $D^{-1}{}_{mm}B_m$ rather than $D^{-1}{}_{mm}$ alone. Suppose $X_m=D^{-1}{}_{mm}B_m$ and $Y_{mf}=D^{-1}{}_{mm}C_{mf}$. Then, to calculate these terms, we need to solve the following linear equations for X_m and Y_{mf}:

$$D_{mm}X_m = B_m \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (13)$$

$$D_{mm}Y_{mf} = C_{mf}. \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (14)$$

B_m is a vector of size $(2n_c+2)n_Hn_v$ and C_{mf} is a matrix of size $(2n_c+2)n_Hn_v$ by $(2n_c+2)$. Hence, by combining these two systems of linear equations, there will be only $(2n_c+3)$ right-hand sides compared to $(2n_c+2)n_Hn_v$ right-hand sides in the first method.

Fluid-Related Calculations Using EOS

Phase-equilibrium calculations play a critical role in both development of an EOS compositional simulator and its efficiency. One of the major convergence problems of EOS compositional simulators is caused by inefficient treatment of the fluid-related calculations. These calculations determine the number, amounts, and compositions of the phases in equilibrium.

With new independent variables, phase-stability analysis based on the tangent plane criterion (Michelsen 1982) is performed only when a gridblock has a single hydrocarbon phase at previous Newton iteration. If the analysis still gives a single phase, phase identification is performed using Gosset et al.'s method, (Gosset et al. 1986), which can be used with no information about equilibrium ratio of each component. If two phases exist, flash calculation is performed giving phase composition and new equilibrium ratio for the next Newton iteration.

When a gridblock has two hydrocarbon phases at previous Newton iteration, both overall mole fractions and equilibrium ratios are updated. In this case, phase number and type can be determined with conventional methods (McCain 1990). If there are two hydrocarbon phases, fugacity coefficient of each component in each phase is calculated based on Peng-Robinson EOS with updated phase composition. Subsequently, following equations which are different forms of Eqs. 5 and 6 are examined to check convergence for phase equilibrium:

$$\ln K_{f\,i} = \ln \Phi^{o}_{f\,i} - \ln \Phi^{g}_{f} \text{ (fracture system) } \ldots\ldots\ldots\ldots\ldots\ldots\ldots\text{(15)}$$

$$\ln K_{mi} = \ln \Phi^{o}_{mi} - \ln \Phi^{g}_{mi} \text{ (matrix system). } \ldots\ldots\ldots\ldots\ldots\ldots\text{(16)}$$

where Φ^{j}_{ki} is the fugacity coefficient of component *i* in phase *j* in media *k* and K_{ki} is the equilibrium ration of component *i* in medium *k,* one of newly updated independent variables.

The key equations used to calculate the fluid properties such as viscosities, relative permeability, capillary pressure, and phase molar density are described in Chang (1990).

Parallel Implementation

Increased oil and gas production from naturally fractured reservoirs using improved oil-recovery processes involves numerical modeling of such processes to minimize the risk involved in development decisions. Production from naturally fractured reservoirs requires more detailed analyses with a greater demand for reservoir simulations with geological and physical models than conventional reservoirs. The computational work required to produce accurate simulations is very high for these problems, and thus there is a great need for parallel computing.

To develop the compositional dual-porosity model in a parallel processing platform, we used a framework approach to handle the complicated tasks associated with parallel processing. The goal was to separate the physical model development from parallel processing development. To achieve this, we employed an Integrated Parallel Accurate Reservoir Simulator (IPARS) framework (Parashar et al. 1997; Wang et al. 1997). The IPARS framework includes a number of advanced features, such as the entire necessary infrastructure for physics modeling, message passing and input/output to solvers and well handling, and the ability to run on a range of platforms from a single PC with Linux Operating System to massively parallel machines or clusters of workstations. The framework allows the representation of heterogeneous reservoirs with variable porosity and permeability and allows the reservoir to consist of one or more fault blocks.

Spatial decomposition of the reservoir gridblocks is applied for parallel processing. The original reservoir simulation domain is divided into several subdomains equal to the number of processors required by the run. The computations are decomposed on parallel machines to achieve computational efficiency. The gridblocks assigned to each processor are surrounded by one or more layers of grid columns representing gridblocks of the neighboring processors. This is referred to as the boundary or ghost layer. The framework provides a routine that updates data in the communication layer. The message passing interface (MPI) is used in the framework to handle necessary message sending/receiving between processors.

The fully implicit EOS compositional formulation has been implemented into the IPARS framework and successfully tested on a variety of computer platforms such as IBM SP and a cluster of PCs (Wang et al. 1999). The goal of this wok was to add a dual-porosity module to the existing compositional Peng-Robinson cubic equation-of-state module. The simulator includes the IPARS framework and the compositional and dual porosity modules. The entire software, which includes the framework and several modules for IOR processes is called General Purpose Adaptive Simulator (GPAS) and is illustrated in **Fig. 3.**

Verification Studies

Numerous verification case studies were performed to test the newly developed dual-porosity option of the GPAS simulator to model naturally fractured reservoirs. Here, we present only a few of them [others can be found in Naimi-Tajdar (2005)]. First, the single-porosity option of the GPAS was tested after implementation of the dual-porosity option using a series of waterflood processes in the fracture and matrix media. Then a modified version of Kazemi et al.'s (1976) quarter five-spot waterflood case (2D and 3D) was used for validation. Finally, the compositional option of the GPAS for naturally fractured reservoirs was verified using a 3D gas-injection process. In all the subsequent cases, the two numbers specified for matrix subgridding are the number of subgrids in horizontal (nested) and vertical (stacked) directions, respectively. For example, 4×1 matrix subgrids means there are four subgrids in horizontal direction and one subgrid in vertical direction (only nested subgrids).

2D Waterflood. A modified Kazemi et al. (1976) quarter five-spot waterflood was used to verify the 2D option of the developed dual-porosity option of GPAS simulator. The results were compared against the results of the UTCHEM simulator (Delshad et al. 1996). Aldejain (1999) has shown "a very close match" between the results of UTCHEM compared against the ECLIPSE simulator (Schlumberger 2004) for a similar case. In this case, water is injected into a quarter-five-spot model at a rate of 200 STB/D and liquids are produced from the other end at a constant pressure of 3900 psia. The reservoir is 600 ft long, 600 ft wide, and 30 ft thick. The fracture media is discretized into 8×8 uniform gridblocks in the x and y directions, respectively, and has one 30-ft-thick gridblock in the z direction. Input parameters are given in **Table 1.** Zero capillary pressure is used for both fracture and matrix media. Relative permeability curves in the fracture and matrix media are shown in **Fig. 4.**

Because ECLIPSE does not have an option to subgrid in the vertical direction, we used a 4×1 subgridding for the matrix media. The oil recovery and oil and water production rates are shown in **Figs. 5 and 6,** respectively. There is excellent agreement between the results of GPAS, UTCHEM, and ECLIPSE. Note that the oil recovery is very low (less than 6% after 1,200 days or 0.625 water-injected pore volumes). Subgridding in the vertical direction was investigated in the next series of runs. **Figs. 7 and 8** show the comparison between the results of GPAS, UTCHEM, and ECLIPSE for cases of 4×1 and 4×4 subgrids. Note that because of the lack of vertical subgridding option in ECLIPSE, there are no results for ECLIPSE run of 4×4 case. A very interesting fact can be noticed by looking closely at these figures. The oil recovery has increased to approximately 20% from 6% with vertical subgridding. Also, the water breakthrough is delayed to approximately 50 days in the 4×4 subgridding case. These results show the importance of vertical subgridding when gravity drainage is an important recovery mechanism from the matrix blocks. Hence, the results of the simulators without vertical subgridding could be very misleading.

To investigate the effect of capillarity in the matrix media, we used the same input parameters of these cases and added the capillary pressure option of GPAS. **Fig. 9** shows the capillary pressure curve used in the matrix media. Again, zero capillary pressure is assumed for the fracture media. **Figs. 10 and 11** show the oil recovery and the oil and water production rates for the 4×4 subgrids case, with and without capillary pressure. There are very interesting features in these figures. First, the oil recovery has increased by a factor of two from 20% to more than 40% after 1,200 days (which equals 0.625 water-injected pore volumes) and second, the water breakthrough has been delayed from 50 days to almost 300 days. These behaviors show that in addition to gravity drainage, we have a very strong imbibition mechanism caused by capillary pressure (Fig. 9) in this case. In other words, water is imbibed into the rock matrix from the fracture and oil comes out of the matrix and is produced through the fracture system. Hence, both gravity drainage and capillary pressure play very important mechanisms in oil recovery in naturally fractured reservoirs.

3D Gas Injection. A six-component, constant-rate, gas-injection process [a modified version of the SPE fifth comparative solution problem (Killough and Kossack 1987)] was used to verify the compositional option of the developed dual porosity module of GPAS. The simulation domain is 560×560×100 ft^3 and is discretized in 7×7×3 fracture gridblocks with the matrix block size of 10×10×10 ft^3 discretized in 2×1 subgrids. The three layers have 20, 30, and 50 ft thickness, respectively, from the top of the reservoir. The initial reservoir pressure is 1,500 psi. The only injection well is located at gridblocks (1,1,1) through (1,1,3) and injects at the constant rate of 1 MMscf/D, and the production well is located at gridblocks (7,7,1) through (7,7,3) and produces at a constant bottomhole pressure of 1,300 psia. The reservoir description of this case is presented in **Table 2.** Straight-lines three-phase relative permeabilities are used for the fracture network and the three-phase relative permeabilities used in matrix media are shown in **Fig. 12.** The hydrocarbon phase behavior is given using a six-component application of the Peng-Robinson equation of state. **Table 3** shows the initial composition and properties of the six components and binary interaction coefficients used in the fracture and matrix media, respectively.

To test the results of the GPAS simulation runs, the dual-porosity option of the GEM simulator (Computer Modeling Group 2004) (compositional mode) was used. Because the GEM simulator did not have an option to discretize the matrix media in compositional mode at the time of this study, we used Kazemi-Gilman's shape factor (Computer Modeling Group 2004) to calculate the matrix-fracture transfer function. The results of the GPAS and GEM simulation runs for oil recovery and oil and gas production rates vs. time are shown in **Figs. 13 through 15,** respectively. The results of the GPAS and GEM showed a good agreement with a slight difference at early time because of the difference in flash calculation between GPAS and GEM.

Parallel Dual-Porosity Reservoir Simulation

The performance of the developed dual-porosity option of GPAS simulations in a parallel processing platform is presented in this section. Speedup ratio is one of the ways to measure parallel processing performance efficiency and is defined as: Speedup = t_1 / t_n, where t_1 is the execution time on a single processor, and t_n is the execution time on n processors. The ideal speedup of parallel simulation with n processors is n, which means the program runs n times faster. However, in reality, as the number of processors becomes larger, a speedup less than n is usually observed. This performance reduction is because of increasing inter-processor communication. Also, it can be caused by an unfavorable programming style, in which a program does not decompose the application evenly (load-balance issue). A Dell PowerEdge 1750 cluster system (Lonestar) was used for these parallel processing cases. The Lonestar cluster system is a Cray-Dell Linux cluster located in the Texas Advanced Computing Center (TACC) of the University of Texas at Austin. The TACC Cray-Dell PowerEdge Xeon Cluster contains 768 Xeon processors at 3.06GHz speed and 256 Xeon processors at 3.2-GHz speed. A Myrinet-2000 switch fabric, employing PCI-X interfaces, interconnects the nodes (I/O and compute) with a point-to-point bandwidth of 250 MB/sec (500 MB/sec bidirectional).

2D Waterflood Case Study. The 2D quarter five-spot waterflood problem was used and scaled up in size to investigate the performance of the developed dual-porosity option of GPAS simulations in a parallel processing platform. The described model used consists of 16,384 gridblocks (4,096 fracture gridblocks and 2×2 matrix subgrids in each fracture gridblock) and two operating wells.

This case was run in serial and parallel modes on the Lonestar cluster system using two, four, eight, and 16 processors. Identical results for oil recovery and water production were obtained for the serial (single-processor) and the parallel (multiprocessors) simulations. **Table 4** lists the speedups and execution times on the Lonestar cluster. **Fig. 16** shows a plot of the speedups for different numbers of processors. As expected, by increasing the number of processors, the execution time for running the case is decreased. The speedups of 3.66 and 6.24 obtained for four and eight processors, respectively, are not too far from the ideal speedup line, whereas the speedup of 10.49 for 16 processors is much farther from the ideal speedup line. The main reason for this is that up to eight processors, the system is large enough for each processor (from a computational point of view). However, the system becomes smaller for each processor using 16 processors, and most of the time is consumed in message passing among the processors. To better understand this issue, the CPU times for major sections of the code such as Linear Solver, Updating Jacobian and Residuals, and calculating Dependent Variables were recorded. These sections were timed individually during the simulation and reported at the end. **Fig. 17** shows pie charts for one and 16 processors, respectively. Note that most of the interprocessor communications are included in the solver calculation. By looking at Fig. 17, it can be seen that the solver takes a large amount of CPU time using 16 processors (28%) compared to one processor (13%), which confirms the increase in the interprocessor communications for the 16 processors run. Also, the CPU time to update the Jacobian matrix using 16 processors (54%) is much smaller than for one processor (67%).

3D Waterflood Case Study. To illustrate the performance of the developed dual-porosity option of GPAS in a parallel processing platform in 3D with more gridblocks (and thus unknowns), simulation of a large 3D five-spot waterflood model with five operating wells was performed. In this case, the total number of gridblocks is 98,304 (64×64×6 fracture gridblocks and 2×2 matrix subgrids in each fracture gridblock). This leads to 393,216 unknowns in the fracture and matrix media solved at each timestep. There are four injection wells injecting at constant rates of 250 STB/D at the corners of the reservoir model and one well producing at a constant bottomhole pressure of 3,500 psia in the center. The initial reservoir pressure in both fracture and matrix media is 4,000 psia.

Because the computational model is large, the simulation was run on the Lonestar cluster system in parallel mode using two, four, eight, 16, and 32 processors. Results of simulation runs show the same oil recovery and oil and water production rates regardless of the number of processors used. **Table 5** lists the speedups and execution times using the Lonestar cluster. **Fig. 18** shows the speedups for different numbers of processors. Again as expected, by increasing the number of processors, the execution time is decreased and the speedup is increased linearly. Fig. 18 shows that the speedup is fairly linear up to 16 processors. However, we have a super linear speedup of 33.4 for 32 processors. One reason for this difference in speedup between the ideal line and 32 processors is probably because processors in the Lonestar cluster system are not homogeneous. As mentioned earlier, in the Lonestar cluster system, some of the processors have 3.06 GHz and some have 3.2 GHz speed, and we have no control on choosing the processors for the simulation runs. However, the chance for the single processor run to be directed to the processors with 3.06 GHz is more than three times that for the processors with 3.2 GHz. This could possibly explain why for the 32 processors run, we exceed the ideal line. **Fig. 19** shows the breakdown of the CPU time for the two-processor run. Results of simulation runs show the same pie chart regardless of the number of processors used. In this case, for a two-processor simulation run, updating the Jacobian takes approximately

65% of the CPU time, while the parallel solver consumes approximately 15% of the computer time.

3D Gas Injection Case Study. The six-component, constant-rate, gas-injection process [a modified version of the SPE fifth comparative solution problem (Killough and Kossack 1987)] was used to investigate the efficiency of the parallel processing simulation of the developed compositional dual porosity module of the GPAS simulator. The simulation domain is 1,280×1,280×100 ft discretized in 64×64×2 fracture gridblocks with the matrix blocks size of 10×10×10 ft discretized in 2×2 subgrids. The two layers each have thickness of 50 ft. In this case, the total number of gridblocks is 32,768 (64×64×2 fracture gridblocks and 2×2 matrix subgrids in each fracture gridblock). This leads to 458,752 unknowns in the fracture and matrix media solved at each timestep. The initial reservoir pressure is 1,500 psia. There are four gas injectors located at corners of the reservoir model that inject at the constant rate of 1 MMscf/D each. The only production well is located in the fracture gridblocks (32,32,1) to (32,32,2) and produces at constant bottomhole pressure of 1,200 psia. The hydrocarbon phase behavior is obtained using a six-component application of the Peng-Robinson equation of state.

Once again, because the computational model is large for this case, there are no results available for the single processor. The simulation case was run on the Lonestar cluster system in parallel mode using two, four, eight, 16, and 32 processors. Results of simulation runs show the same oil recovery and oil and gas production rates regardless of the number of processors used. **Table 6** lists the speedups and execution times on the Lonestar cluster. **Fig. 20** shows plot of the speedups for different number of processors and indicates a linear speedup for up to 32 processors.

In this large-scale simulation, the largest computational tasks were updating the Jacobian and the linear solver, which account for more than 90% of the total execution time, as seen in the pie chart for the 32-processors run **(Fig. 21).** Uetani et al. (2002) reported approximately 75% and Dogru et al. (1999) reported approximately 76% of the CPU time is consumed in these two calculations for single-porosity parallel simulators, which is very similar to the case reported in this study. However, there is a very important difference between this case and the other two. Based on Uetani et al.'s paper, of the 75%, the parallel solver consumes more than 44% while updating Jacobian consumes about 31%. Also, based on Dogru et al.'s paper, the parallel solver consumes 61% and updating the Jacobian 15% of the computation time. Hence, the biggest chunk of CPU time consumption in single-porosity parallel simulators is the parallel solver. By looking at Fig. 21, the parallel solver consumes only approximately 20% of the computer time while updating the Jacobian consumes more than 70%. The main reason for this difference is the nature of the solution approach in the dual-porosity option of GPAS. As discussed in Solution Approach section of this paper, the entire calculation for the matrix unknowns is done locally within the processors, while calculations for the fracture unknowns are done in the parallel solver. Hence, in our model, most of the calculations are done locally and we do not need much interprocessor communications. This fact confirms the suitability of the developed dual-porosity option of GPAS in parallel processing platforms. The other reason is that this is a compositional run with six components. Flash calculations, which include updating the Jacobian and residuals are calculated locally, and are very time consuming. This explains the increase of almost 10% in updating the Jacobian in this case relative to the 3D waterflood case.

Conclusions

The following conclusions are drawn from this study:

1. A new, parallel, equation-of-state compositional, fully implicit simulator to model naturally fractured oil reservoirs has been developed and verified successfully against the UTCHEM

simulator as well as other commercial simulators. The methodology used to develop this reservoir simulator is based on a modified dual-porosity model.

2. The matrix blocks were discretized in the horizontal direction using a modified multiple interacting continua (MINC) method concept and in the vertical direction using a stacked grid concept to provide an accurate calculation for the matrix-fracture transfer function. To the best of our knowledge, there is currently no compositional reservoir simulator that has the capability of subgridding in both horizontal and vertical directions. This subgrid scheme was used to model 3D flow in the matrix rock with 2D subgrids with good accuracy.
3. The efficiency of the parallel processing of GPASv3.5 was verified using 2D and 3D waterflood as well as 3D compositional gas-injection processes. For the 2D waterflood process, a linear speedup up to eight processors was achieved. For the 3D waterflood process, a linear speedup was obtained using up to 32 processors. In the 3D compositional gas injection with six hydrocarbon components cases, because of the large size of the problem, excellent speedup was achieved. In this case, the linear solver and updating the Jacobian accounted for the majority of the CPU time consumption. Flash and phase behavior calculations in the fracture gridblocks and matrix subgrids, which are performed locally, account for this increase of CPU time consumption in updating the Jacobian.
4. The efficiency of parallel processing using the developed dual-porosity simulator was better than the parallel efficiency of our single-porosity simulator. The main reason is that based on the solution approach used in the dual-porosity simulator, the unknowns in the matrix media are solved locally in each processor that contains the corresponding fracture gridblock. Therefore, there would be much less time consumed in inter-processor communication that arises in the linear solver, thereby increasing the parallel efficiency of the developed dual-porosity simulator.

Nomenclature

D =	depth measured positive downward, L
f =	fugacity
K_i =	equilibrium ratio, dimensionless
L_j =	mole fraction of phase j
n_c =	number of hydrocarbon components
n_p =	number of phases
N_b =	number of matrix subgrids
N_i =	moles of component i per unit pore volume, mol/L^3
NM =	number of matrix blocks within a fracture gridblock
P =	pressure, m/Lt^2
q =	flow rate, mol/t
t =	time, t
V =	volume, L^3
x_{ij} =	mole fraction of component i in phase j
γ =	fluid specific gravity m/L^2t^2
λ =	effective mobility, L^3t/m
$\bar{v}_j$ =	molar volume of phase j
ξ_j =	molar density of phase j, mol/L^3
τ =	matrix-fracture transfer function, mol/t
ϕ =	porosity, fraction
Φ =	fugacity coefficient, fraction

Subscripts

b =	bulk
f =	fracture
g =	gas
i =	component index
j =	phase index
m =	matrix

o = oil

w = water

Acknowledgments

The financial support of the Reservoir Simulation Joint Industry Project in the Center for Petroleum and Geosystems Engineering at the University of Texas at Austin is gratefully acknowledged. Todd Arbogast was supported by the NSF under grant DMS-0417431.

References

1. Aldejain, A.A. 1999. Implementation of a Dual Porosity in a Chemical Flooding Simulator. PhD dissertation. Austin, Texas: University of Texas.
2. Baker, L.E., Pierce, A.C., and Luks, K.D. 1982. Gibbs Energy Analysis of Phase Equilibria. *SPEJ* **22** (5): 731–742. SPE-9806-PA. DOI: 10.2118/9806-PA.
3. Barenblatt, G.E., Zheltov, I.P., and Kochina, I.N. 1960. Basic Concepts in the Theory of Seepage of Homogeneous Liquids in Fissured Rocks. *J. Applied Mathematics USSR* **24** (5): 1286–1303.
4. Beckner, B.L., Firoozabadi, A., and Aziz, K. 1987. Modeling Transverse Imbibition in Double-Porosity Simulators. Paper SPE 17414 presented at the SPE California Regional Meeting, Long Beach, California, 23–25 March. DOI: 10.2118/17414-MS.
5. Beckner, B.L., Chan, H.M., McDonald, A.E., Wooten, S.O., and Jones, T.A. 1991. Simulating Naturally Fractured Reservoirs Using a Subdomain Method. Paper SPE 21241 presented at the SPE Symposium on Reservoir Simulation, Anaheim, California, 17–20 February. DOI: 10.2118/21241-MS.
6. Chang, Y.-B. 1990. Development and Application of an Equation of State Compositional Simulator. PhD dissertation. Austin, Texas: University of Texas.
7. Chen, W.H., Wasserman, M.L., and Fitzmorris, R.E. 1987. A Thermal Simulator for Naturally Fractured Reservoirs. Paper SPE 16008 presented at the SPE Symposium on Reservoir Simulation, San Antonio, Texas, 1–4 February. DOI: 10.2118/16008-MS.
8. Computer Modeling Group Ltd. 2004. *GEM Users Guide.*
9. Dean, R.H. and Lo, L.L. 1988. Simulations of Naturally Fractured Reservoirs. *SPERE* **3** (1): 638–648; *Trans.,* AIME, **285.** SPE-14110-PA. DOI: 10.2118/14110-PA.
10. Delshad, M., Pope, G.A., and Sepehrnoori, K. 1996. A Compositional Simulator for Modeling Surfactant Enhanced Aquifer Remediation, 1 Formulation. *J. Contamin. Hydrol.* **23** (4): 303–327.
11. Dogru, A.H. et al. 1999. A Massively Parallel Reservoir Simulator for Large Scale Reservoir Simulation. Paper SPE 51886 presented at the SPE Reservoir Simulation Symposium, Houston, 14–17 February. DOI: 10.2118/51886-MS.
12. Douglas, J. Jr., Arbogast, T., and Paes Leme, P.J. 1989. Two Models for the Waterflooding of Naturally Fractured Reservoirs. Paper SPE 18425 presented at the SPE Symposium on Reservoir Simulation, Houston, 6–8 February. DOI: 10.2118/18425-MS.
13. Gilman, J.R. 1986. An Efficient Finite-Difference Method for Simulating Phase Segregation in Matrix Blocks in Double-Porosity Reservoirs. *SPERE* **5** (6): 403–413; *Trans.,* AIME, **281.** SPE-12271-PA. DOI: 10.2118/12271-PA.
14. Gilman, J.R. and Kazemi, H. 1983. Improvements in Simulation of Naturally Fractured Reservoirs. *SPEJ* **23** (4): 695–707. SPE-10511-PA. DOI: 10.2118/10511-PA.
15. Gosset, R., Heyen, G., and Kalitventzeff, B. 1986. An Efficient Algorithm to Solve Cubic Equations of State. *Fluid Phase Equilibria* **25** (1): 51–64.
16. Kazemi, H., Merrill, L.S. Jr., Porterfield, K.L., and Zeman, P.R. 1976. Numerical Simulation of Water-Oil Flow in Naturally

Fractured Reservoirs. *SPEJ* **16** (6): 317–326; *Trans.,* AIME, **261.** SPE-5719-PA. DOI: 10.2118/5719-PA.

17. Killough, J.E. and Kossack, C.A. 1987. Fifth Comparative Simulation Project: Evaluation of Miscible Flood Simulations. Paper SPE 16000 presented at the SPE Symposium on Reservoir Simulation, San Antonio, Texas, 1–4 February. DOI: 10.2118/16000-MS.
18. Lake. L.W. 1989. *Enhanced Oil Recovery*. Englewood Cliffs, New Jersey: Prentice Hall.
19. McCain, W.D. Jr. 1990. *The Properties of Petroleum Fluids*. Tulsa: PennWell Books.
20. Michelsen, M.L. 1982. The Isothermal Flash Problem. Part I. Stability. *Fluid Phase Equilibria* **9** (1): 1–19.
21. Naimi-Tajdar, R. 2005. Development and Implementation of a Naturally Fractured Reservoir Model Into a Fully Implicit, Equation-of-State Compositional, Parallel Simulator. PhD dissertation. Austin, Texas: University of Texas.
22. Parashar, M., Wheeler, J.A., Pope, G., Wang, K., and Wang, P. 1997. A New Generation EOS Compositional Reservoir Simulator: Part II—Framework and Multiprocessing. Paper SPE 37977 presented at the SPE Reservoir Simulation Symposium, Dallas, 8–11 June. DOI: 10.2118/37977-MS.
23. Pruess, K. and Narasimhan, T.N. 1985. A Practical Method for Modeling Fluid and Heat Flow in Fractured Porous Media. *SPEJ* **25** (1): 14–26. SPE-10509-PA. DOI: 10.2118/10509-PA.
24. Rossen, R.H. and Shen, E.I.C. 1989. Simulation of Gas/Oil Drainage and Water/Oil Imbibition in Naturally Fractured Reservoirs. *SPERE* **4** (4): 464–470; *Trans.,* AIME, **287.** SPE-16982-PA. DOI: 10.2118/16982-PA.
25. Saidi, M.A. 1975. Mathematical Simulation Model Describing Iranian Fractured Reservoirs and Its Application to Haft Kel Field. *Proc.,* Ninth World Petroleum Congress, Tokyo 13 (3): 209–219.
26. Schlumberger Ltd. 2004. ECLIPSE Technical Description.
27. Thomas, L.K., Dixon, T.N., and Pierson, R.G. 1983. Fractured Reservoir Simulation. *SPEJ* **23** (1): 42–54. DOI: 10.2118/9305-MS.
28. Uetani, T., Guler, B., and Sepehrnoori, K 2002. Reservoir Simulation on High Performance Clusters. Paper presented at the Sixth World Multiconference on Systematics, Cybernetics, and Informatics, Orlando, Florida, 14–18 July.
29. Wang, P. et al. 1997. A New Generation EOS Compositional Reservoir Simulation: Part I—Formulation and Discretization. Paper SPE 37979 presented at the SPE Reservoir Simulation Symposium, Dallas, 8–11 June. DOI: 10.2118/37979-MS.
30. Wang, P. et al. 1999. A Fully Implicit Parallel EOS Compositional Simulator for Large Scale Reservoir Simulation. Paper SPE 51885 presented at the SPE Reservoir Simulation Symposium, Houston, 14–17 February. DOI: 10.2118/51885-MS.
31. Warren, J.E. and Root, P.J. 1963. The Behavior of Naturally Fractured Reservoirs. *SPEJ* **3** (3): 245–255; *Trans.,* AIME, **228.** SPE-426-PA. DOI: 10.2118/426-PA.
32. Wu, Y.-S. and Pruess, K. 1988. A Multiple-Porosity Method for Simulation of Naturally Fractured Petroleum Reservoirs. *SPERE* **3** (1): 327–336; *Trans.,* AIME, **285.** SPE-15129-PA. DOI: 10.2118/15129-PA.

SI Metric Conversion Factors

Ft	×	3.048	E–01 = m
psi	×	6.894 757	E+00 = kPa

Bio

Figure Legends

Fig. 1—Idealized representation of fractured reservoirs (Warren and

Root 1963).

Fig. 2—Basic 2D computational mesh for fractured reservoir (Pruess and Narasimhan 1985).

Fig. 3—General schematic of GPAS.

Fig. 4—Fracture and matrix relative permeabilities used in the quarter-five-spot waterflood verification study.

Fig. 5—Oil recovery vs. time for a quarter five-spot waterflood (4×1 subgrids).

Fig. 6—Oil and water production rates for a quarter five-spot waterflood (4×1 subgrids).

Fig. 7—Oil recovery for a quarter five-spot waterflood (4×1 and 4×4 subgrids).

Fig. 8—Oil and water production rates for a quarter five-spot waterflood (4×1 and 4×4 subgrids).

Fig. 9—Imbibition capillary pressure curve used in the quarter five-spot waterflood (4×4 subgrids).

Fig. 10—Oil recovery for the quarter five-spot waterflood (4×4 subgrids, with and without capillary pressure).

Fig. 11—Oil and water production rates for the quarter five-spot waterflood (4×4 subgrids, with and without capillary pressure).

Fig. 12—Three-phase matrix relative permeabilities used in the gas injection.

Fig. 13—Oil recovery vs. time for gas-injection validation.

Fig. 14—Oil production rate vs. time for gas-injection validation.

Fig. 15—Gas production rate vs. time for gas-injection validation.

Fig. 16—Speedups for parallel simulations (2D waterflood).

Fig. 17—Execution time breakdown for 2D waterflood (1 and 16 processors).

Fig. 18—Speedups for parallel simulations (3D waterflood).

Fig. 19—Execution time breakdown for 3D waterflood (2 processors).

Fig. 20—Speedups for parallel simulations (gas injection).

Fig. 21—Execution time breakdown for gas injection (32 processors).