Parallel Finite Element Simulation of Tracer Injection in Oil Reservoirs

Alvaro L.G.A. Coutinho and José L.D. Alves
Center for Parallel Computations, COPPE/Federal University of Rio de Janeiro
P.O. Box 68506, Rio de Janeiro, RJ 21945-970, Brazil
alvaro@coc.ufrj.br , jalves@coc.ufrj.br

ABSTRACT

In this work, parallel finite element techniques for the simulation of tracer injection in oil reservoirs are addressed. The pressure equation is approximated by Galerkin's method and the velocity field computed through a post-processing approach to recover the required accuracy. The concentration equation is approximated in space by the Streamline Upwind Petrov Galerkin (SUPG) plus a discontinuity-capturing operator. The resulting semi-discrete equations are approximated in time by a predictor-multicorrector algorithm. The pressure, velocity and concentration linear systems of equations are solved with parallel element-by-element iterative techniques. Performance measurements on the CRAY YMP and the CRAY C90 for the injection of tracer on a five-spot pattern with random small scale permeability variations are performed to demonstrate that the numerical techniques employed are accurate and result in a fast code.

INTRODUCTION

The numerical simulation of processes for hydrocarbon recovery is one of the most computational intensive engineering activities. Supercomputers have made it possible to consider global reservoir effects which can not be represented using crossections, average patterns or small reservoir segments. High resolution models have been used for detailed simulations of viscous fingering and small scale heterogeneities. Standard reservoir simulators are based on finite differences approximations in time and space and nowadays are designed to achieve high performance in today's parallel supercomputers (Young and Hemanth-Kumar[1,2]). Calculation rates of 1.2 *Gflops* for black oil simulations and 1.8 *Gflops* fos EOS simulations were achieved on a 16 CPU's C90. However, in the simulation of tracer injection usually standard codes do not work well. To model the flow of small concentrations of chemical or radioactive tracers on the reservoir, the usual upwind finite difference approximation and cartesian grids do not provide the required resolution.

In recent years there have been a renewed interest on the utilization of finite element approximations in reservoir simulations, mainly by its ability to handle complex reservoir geometries, discontinuities (such as faults), to improve the resolution around the wells (Fung et al [3]), and to model tracer injection (Loula et al[4]). However, to achieve the high performance observed in the finite difference simulators special attention should be given to the finite element solution techniques employed. Differently from finite differences, the most important computational kernel of finite element analysis is the solution of the systems of linear equations, basically by the unstructured nature of finite element grids.

For most problems of practical interest, especially in three-dimensions, solution methods based on direct methods, such as Gauss elimination, lead to massive storage demands and large computer times. Iterative methods, on the other hand, present comparatively low storage requirements and, when associated with suitable preconditioners, provide a powerful computational strategy. In this work we employ iterative techniques in combination with element-by-element solution strategies. In these strategies the residual and preconditioning computations are kept at element level. Consequently, the needs for the formation, storage and factorization of large global matrices are eliminated. Further, by the application of an element reordering algorithm suitable for arbitrary meshes, the element-by-element strategies can be vectorized and parallelized, thus achieving the desired high performance. For a recent review of such topics, please see Tezduyar et al[5], Behr and Tezduyar[6] , and Coutinho et al[7]. To further enhance the overall computational efficiency, we also employ an adaptive time step size control strategy.

In the next Section we briefly review the mathematical model and present the resulting finite element equations. The Section that follows describes the parallel solution techniques. Two examples were selected. The first one is the injection of a tracer on a five spot configuration on a homogeneous media and other one considers a random heterogeneous media. Parallel performance analyses were accomplished for this example on two different machines, a CRAY YMP-2E and a CRAY C98-E. Finally, we gather the main conclusions of this work in the last Section.

MATHEMATICAL MODEL AND FINITE ELEMENT EQUATIONS

According to Chavent and Jaffre[8], the flow in a porous medium Ω with boundary $\partial\Omega$ of two miscible incompressive fluids (for instance, tracer and water), in a time interval $[0 \times T]$, can be described neglecting gravity, by the set of partial differential equations

$$\nabla . v = q_1(x,t) \tag{1}$$

$$v = -A(u,x).\nabla p \tag{2}$$

$$\phi \frac{\partial u}{\partial t} + \nabla .(uv) - \nabla .(D\nabla u) = q_2(x,t) \tag{3}$$

where the spatial and temporal coordinates are denoted respectively by x and t, ϕ is the porosity, v is the total Darcy velocity, p the total fluid pressure, u is the tracer concentration. The wells are represented by the source terms q_1 and q_2. We assume the usual *no-flow* boundary conditions

$$v.n = 0 \qquad \text{in } \partial\Omega \times [0 \times T] \tag{4}$$

where n is the unit outward normal. A proper initial condition for u is given. The particular forms of tensors A and D are given in the Appendix. Substituting Eq.(2) into (1), we have the pressure equation, that reads,

$$\nabla .(A(u,x).\nabla p) + q_1(x,t) = 0 \tag{5}$$

which is approximated by a standard Galerkin formulation,i.e.,

$$\int_\Omega \nabla w^h A(u^h,x) \nabla p^h d\Omega = \int_\Omega w^h q_1 d\Omega \tag{6}$$

where w is the discrete weighting function and p^h, u^h are the discrete counterparts of p and u. The functions w^h, p^h and u^h are defined over the usual finite element spaces. The concentration equation is approximated by the following stabilized variational formulation,

$$B(w^h,u^h)+\sum_{e=1}^{Nel}\int_{\Omega} v.\nabla w^h \tau_1(Lu^h-q_2)d\Omega$$
$$+\sum_{e=1}^{Nel}\int_{\Omega^e}\frac{\nabla u^h}{|\nabla u^h|}\nabla w^h \tau_2(Lu^h-q_2)d\Omega = L(w^h) \qquad (7)$$

where *Nel* is the total number of elements in the grid and,

$$B(w^h,u^h)=\int_{\Omega} w^h Lu^h d\Omega \qquad (8)$$

$$L(w^h)=\int_{\Omega} w^h q_2 d\Omega \qquad (9)$$

correspond, after the usual integration by parts, to the Galerkin term. The second integral in Eq.(7) is the Streamline Upwind Petrov-Galerkin (SUPG) term (Brooks and Hughes[9]) and the last one is the discontinuity-capturing term (Hughes,Mallet and Mizukami[10]). The operator $L\,u^h$ is given by,

$$Lu^h=\phi\frac{\partial u}{\partial t}+\nabla.(u^h v^h)-\nabla.(D\nabla u^h) \qquad (10)$$

The above formulation is variationally consistent, in the sense that if $u^h \rightarrow u$, the stabilization terms vanish. The discontinuity-capturing term is a nonlinear operator designed to control oscillations around sharp fronts and discontinuities, retaining the higher-order accuracy in smooth regions. Parameters τ_1 and τ_2 are designed to provide the correct amount of diffusion respectively along the streamlines and along the sharp solution gradients.

It is well know that computing velocities directly from Darcy's law yields a low accuracy filed, satisfying only weakly the no-flow boundary conditions. To improve the quality of the velocity approximation we use here the post-processing technique from Loula et al[4] that can be summarized as follows. Given p^h, the Galerkin solution of the pressure equation, the post-processed velocities v_h^p are obtained from the following variational statement,

$$\int_{\Omega} w_v^h(A^{-1}v_p^h+\nabla p^h)d\Omega+M^{1/2}\sum_{e=1}^{Nel}\int_{\Omega}\delta_e\nabla.w_v^h(v_p^h-q_1)d\Omega=0 \qquad (11)$$

where δ_e is a mesh dependent parameter of magnitude $O(h)$. The resulting equations in matrix form corresponding respectively to the pressure equation, the post-processing of the velocities and the concentration equation are :

$$\mathbf{Kp}=\mathbf{Q} \qquad \textit{(Block 1)} \qquad (12)$$

$$\overline{\mathbf{M}}\mathbf{v}=\mathbf{F_G}+\mathbf{F_q} \qquad \textit{(Block 2)} \qquad (13)$$

$$\widetilde{\mathbf{M}}\dot{\mathbf{u}}+\widetilde{\mathbf{C}}\mathbf{u}=\widetilde{\mathbf{F}} \qquad \textit{(Block 3)} \qquad (14)$$

where **p** is the vector of unknown nodal values of p^h , **v** is the vector of unknown nodal post-processed velocities, v_i^p, and **u** is the unknown nodal concentration vector corresponding to u^h . The superimposed dot stands for derivation with respect to time. The matrix **K** is symmetric and positive definite (SPD). Vector **Q** accounts for the flow rates and boundary conditions for the pressure equation. The matrix $\overline{\mathbf{M}}$, also SPD, can be splitted in the sum of two matrices,

$$\overline{\mathbf{M}} = \mathbf{M}_w + \mathbf{M}_{div} \tag{15}$$

where $\mathbf{M}_w$ represents the contribution of the first integral of Eq.(11) and $\mathbf{M}_{div}$ the part corresponding to the divergence weighting of the second integral of the same equation. The vectors $\mathbf{F}_G$ and $\mathbf{F}_q$ are respectively the weighting of the pressure gradients and the divergence weighting of the flow rates. The matrices $\widetilde{\mathbf{M}}$ and $\widetilde{\mathbf{C}}$ are derived from the time dependent and convective-diffusive terms of the concentration equation. The vector $\widetilde{\mathbf{F}}$ is due to the boundary conditions. The arrays with superimposed tilde can be decomposed into their Galerkin, SUPG and discontinuity-capturing parts:

$$\widetilde{\mathbf{M}} = \mathbf{M} + \mathbf{M}_{PG} \tag{16}$$

$$\widetilde{\mathbf{C}} = \mathbf{C} + \mathbf{C}_{PG} + \mathbf{C}_{DC} \tag{17}$$

$$\widetilde{\mathbf{F}} = \mathbf{F} + \mathbf{F}_{PG} \tag{18}$$

where the subscripts PG and DC identify respectively the SUPG and discontinuity capturing contributions. It should be noted that matrices $\widetilde{\mathbf{M}}$ and $\widetilde{\mathbf{C}}$ are non-symmetric. An implicit time discretization is accomplished by the generalized trapezoidal rule (Hughes[12]) and employs a predictor-multicorrector algorithm. The solution advances in time solving sequentially each one of the above three linear system of equations (Eqs.11-13).

The major computational tasks of the present scheme are the generation of the coefficient matrices and right-hand-side vectors for the three blocks, and the corresponding solution of the linear systems of equations. Considering linear triangles, element matrix computations can be carried out in closed form, in a very compact and elegant fashion. Thus all element matrices can be evaluated in single `DO LOOPS`, which can be fully vectorized and parallelized. The resulting matrices are stored element wise, favoring the solution strategies employed presented herein, as we shall see in the next Section. The pressure equation iterative driver is the conjugate gradient method with a symmetrized EBE Gauss-Seidel preconditioner. The velocities are computed using EBE Jacobi iterations. The non-symmetric systems of equations corresponding to the concentration equation are solved by GMRES(k) algorithm, also employing an EBE Gauss-Seidel preconditioner.

To increase the robustness of our solution scheme a control over the time increment is necessary. The complexity of our problem certainly prevents us from developing a complete theory for this case. We adopted here the approach advocated by the feedback control theory (Gustafsson[12]). An estimate of the solution error is compared with the user specified accuracy requirement (TOL) and the result is fed back and used to determine the new stepsize. The controller should keep the error close to the required tolerance. In this work we adopted a PID controller.

PARALLEL ELEMENT-BY-ELEMENT TECHNIQUES

In the scheme outlined above, three systems of equations should be solved at every iteration. The objective of this Section is to present techniques to minimize the computational cost associated to the solution of these systems. After dropping all the subscripts, Equations (11) to (13) can be rewritten in the general form,

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{19}$$

where **A** is the coefficient matrix , **b** is the known right-hand side vector and **x** is the solution sought. For the sake of completeness, we briefly describe the element-by-element (EBE) iterative techniques. In this technique the elements are arranged into N_g groups such that no element within a group share a common node (or degree-of-freedom). In this way, element-by-element (EBE) computations within each group can be vectorized and parallelized (Behr and Tezduyar[6],Coutinho et al[7]). The overhead associated to the element grouping is minimum and it is performed before starting the computations. Based on the grouping, the matrix **A** can be rewritten as,

$$\mathbf{A} = \sum_{k=1}^{N_g} \mathbf{A}_k \tag{20}$$

and the group matrices are defined as,

$$\mathbf{A}_k = \sum_{e \in E_k} \mathbf{A}_e \tag{21}$$

where E_k is the set of elements belonging to group k. For the solution of *Block 1*, we employ the EBE technique in conjunction with the preconditioned conjugate gradient method (PCG). In *Block 2*, we perform EBE Jacobi iterations, while in *Block 3*, the GMRES(k) algorithm is employed, also combined with the EBE technique. The matrix-vector multiplications needed in PCG, Jacobi and GMRES(k) algorithms are computed at element level. An EBE Gauss-Seidel preconditioner is employed for PCG and GMRES(k) algorithms (Shakib et al[13]). Since the off-diagonal components of the Gauss-Seidel factors are identical to the scaled element matrices either in the symmetric or non-symmetric cases, no additional storage areas besides the element matrices are needed. Therefore, all the preconditioning computations are kept at element level and are performed within the different groups. It is important to note that costly assembly operations are completely avoided in the EBE technique. Storage requirements are directly proportional to the number of elements in the mesh, which is very convenient for large-scale problems.

NUMERICAL EXAMPLES

Two examples were selected. The first one is the injection of a tracer on a five spot configuration on a homogeneous media. The purpose of this example is to validate the finite element formulation presented herein. The other one considers a random heterogeneous media. Parallel performance analyses were accomplished for this example, trying to follow as much as possible the guidelines from Crowl[14], on two different machines, a CRAY YMP-2E and a CRAY C98-E. Both machines were non-dedicated. As we shall see, the analyses involve very fine grids, so we did not measure the variance in elapsed time. We will present performance measurements for some important inner

kernels and for the whole jobs. This had to be done due to the different algorithms employed and to account for some extra burdens of the actual runs, as generation of visualization files. The basic variables for the performance measurements are *flops* rate and *speed-up* factor. Although these variables can be subject of criticism, they are accessed using the performance measurement tools available in the machines, *hpm*, *perfview*, and *atexpert*. The *flops* rates are evaluated using the hardware performance monitor (*hpm*) and the profiling tool (*perfview*) on a single CPU version of the code, where vectorization is the major issue. Parallel speed-up's are evaluated by *atexpert* tool. Actual data for parallel performance were only available respectively for 2-CPU's of a CRAY YMP and 8-CPU's of a CRAY C90. All other data were extrapolations performed by *atexpert*. Parallel *flops* were obtained from the scaled rates of the single CPU version of the code to avoid the well known parallel *flop* overhead.

Consider first the injection of a tracer slug of 5% pore volumes on a quarter of a five-spot, with the side of the corresponding computational domain being 1000 ft. The material data are those found in Douglas et al[15]. We suppose initially a homogeneous media with unit mobility ratio, permeability 100 mD, porosity 0.1, longitudinal and transversal dispersivities respectively equal to 4.0 ft and 1.0 ft. To validate our code, Fig. 1 shows the concentration of tracer at the producer for various meshes, in comparison with the analytical solution of Abbaszadeh and Brigham[16].

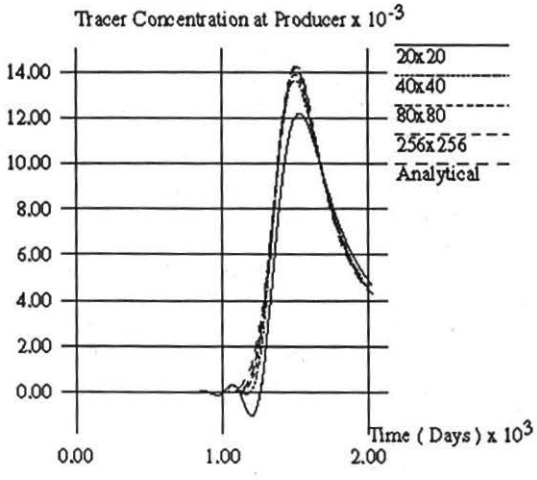

Fig. 1 - Convergence study of the finite element solution.

We can see that for increasingly refined meshes the finite element solutions converge to the analytical solution. It should be noted here that since we are using linear triangles, the number of finite elements is twice the number of cells, i.e., for the 40x40 mesh we have 1600 cells and 3200 triangular elements. The next analysis, performed on the finer mesh (256x256), takes into consideration the effects of a small-scale permeability

variation. For each finite element in the grid, a different value of permeability was generated assuming a uncorrelated random lognormal distribution with mean 100 mD, and coefficient of variation , 1.6 . This yields a maximum value of permeability of 10,956 mD and a minimum of 0.0218 mD, which reflects the degree of heterogeneity of the simulated porous media. Considering a simulation time of 3 pore volumes injected, or 6000 days, Fig. 2 shows a comparison between the homogeneous and heterogeneous numerical solutions.

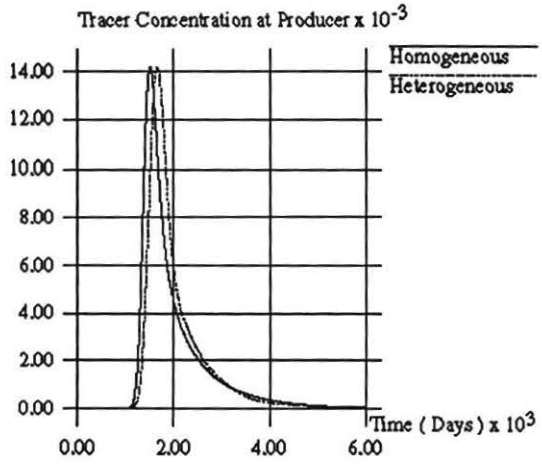

Fig.2 - Finite element solutions for the homogeneous and heterogeneous porous media.

In Fig. 3 the concentration distribution (light gray areas) at time t=500 days is depicted, where is possible to appreciate the effects of the heterogeneous media on the flow pattern.

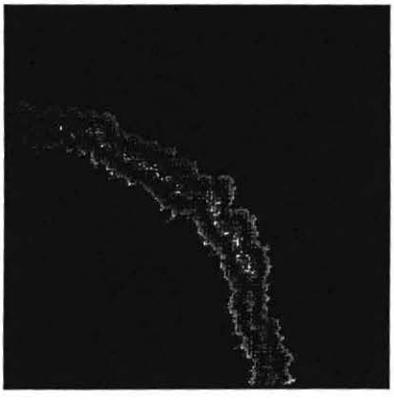

Fig.3 - Concentration map at t=500 days.

In this type of analysis the interaction between pressure/velocity and concentration is disregarded. Therefore, the pressure equation solution and velocity post-processing were performed just once, and only the concentration equation was actually solved at each time step. Table 1 summarizes the main computational data for this analysis.

Number of Elements	131,072
Number of Nodes	66,049
Iterative Solvers Tolerance	10^{-6}
PCG Iterations	1,071
Jacobi Iterations	38
Number of Steps	1,943
GMRES Iterations/Time Step	19.12

Table 1. Computational data for the problem of tracer injection on random heterogeneous media.

The CPU time of the vectorized single processor run on the YMP is 2,845 sec , and for the C90, 1,070 sec. The top 5 routines on CPU utilization, for both machines, are shown in Table 2 below. In this Table *matvec* stands for the non-symmetric matrix-vector multiplication needed in GMRES; *tria2d* is the generation of the concentration equation finite element matrices; *fwds* and *bkds* are the forward and backward solutions necessary to build the non-symmetric EBE Gauss-Seidel preconditioner; *gmres* stands for the non-symmetric iterative driver. As we can see, solution costs are dominated by the matrix-vector multiplications and preconditioner solutions. However, the generation of element matrices is not inexpensive.

Routine	**CPU YMP(%)**	**CPU C90(%)**
matvec	33.9	34.2
tria2d	17.9	16.8
fwds	17.0	16.8
bkds	15.2	13.4
gmres	7.6	8.0
others	8.4	7.4

Table 2 - CPU profile of vectorized single processor runs.

Parallel speed-up's on the YMP for the most important inner kernels listed above and for the whole job are presented in Fig. 4. This Figure also shows the perfect speed-

up curve. We may note the excellent parallel performance of *matvec* and *tria2d*. However, the kernels for the preconditioner construction were not so effective.

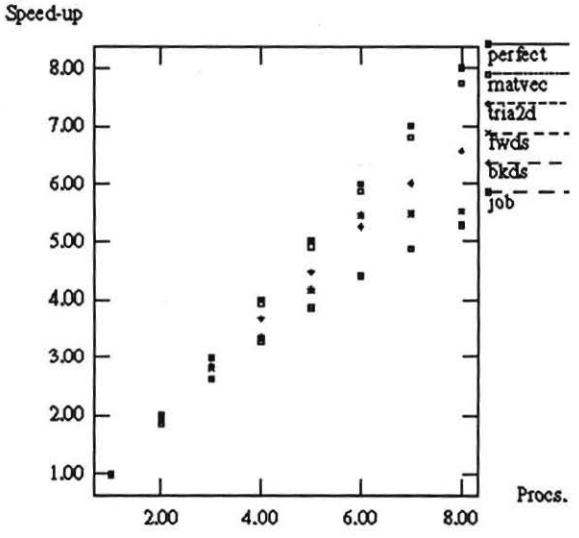

Fig.4 - Parallel *speed-up*'s on the YMP.

The whole job, which includes several other tasks, presented a good parallel performance. Among the tasks that contribute to decrease the job's speed-up, the generation of the visualization files cannot be disregarded. Although very fast, output is not parallel and one file was dumped to disk every 100 time step. Table 3 presents a summary of the *Mflops* rates for the single CPU and parallel execution runs.

Routines	**1-CPU (Mflops)**	**Parallel Speed-up**	**8-CPU's (Mflops)**
matvec	123	7.74	949
tria2d	202	6.57	1,326
fwds	104	5.51	575
bkds	117	5.52	643
job	137	5.28	724

Table 3 - Performance summary on the YMP for the tracer injection on random heterogeneous media.

Comparisons with well-documented performance data of these algorithms on the YMP are not easy. We observed that the conjugate gradient benchmark (class A) from NAS [17] runs at about 127 *Mflops* on one CPU of a YMP, with a parallel speed-up on 8-processors of 5.01. Most of our code, like this benchmark, is dominated by sparse matrix-vector operations. However, several differences are important to point out. In our code we use EBE techniques with mesh coloring; this involves many gather-scatter operations; vectors usually are as long as the size of each color; the non-symmetric driver, GMRES, is far more complex than conjugate gradients. Other interesting performance measurements are available for an unstructured grid finite volume Euler solver[18], which has a kernel akin to our coefficient generation routine, *tria2d*. This code runs at 150 *Mflops* in a YMP. With all these arguments in mind, we are confident to state that our code achieved a very good performance on the YMP.

Parallel speed-up's for the C90 are shown in Fig. 5. Although results up to 8-CPU's are very good, the speed-up's curves tend to flatten for higher number of processors, with the exception of *matvec*.

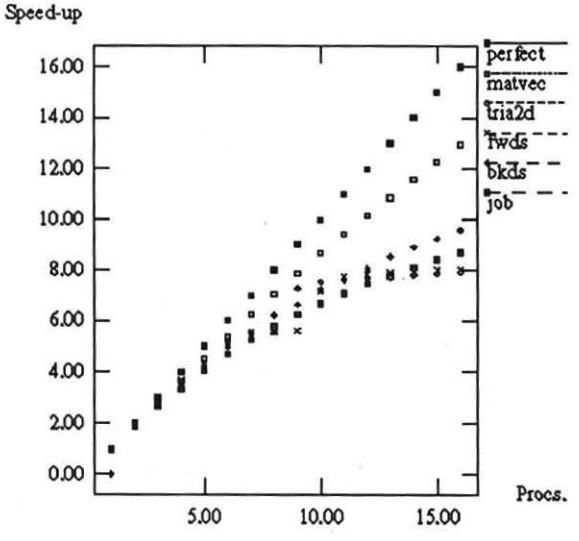

Fig.5 - Parallel *speed-up*'s on the C90.

Table 4 lists the performance summary for the C90. As we can see, single CPU's *Mflops* are within the range of 2.6 to 2.8 times those observed for the YMP. This is in close agreement with the available data from the above mentioned NAS benchmark. Also, for this benchmark, the reported parallel speed-up in a 16 CPU's C90 is 7.86. Parallel speed-up's are not as good as we found for the YMP, but the extrapolated job's performance for a 16-CPU's C90 is still impressive. We stress here that the efficiency decrease on the C90 is mostly due to problem's size. We have 131,072 finite elements in the mesh, but the average color length is 16,384, which makes the effective size of this problem much smaller.

Routines	1-CPU (Mflops)	Parallel Speed-up	16-CPU's (Gflops)
matvec	323	12.94	4.18
tria2d	574	9.58	5.50
fwds	234	8.04	1.88
bkds	351	7.92	2.78
job	365	8.69	3.18

Table 4 - Performance summary on the C90 for the tracer injection on random heterogeneous media.

CONCLUSIONS

In this paper we have presented solution techniques for the equations resulting from a finite element formulation for the flow of tracers on incompressive fluids. The numerical experiments performed clearly show the potentiality of the proposed approach. A parallel performance analysis for a large-scale tracer injection simulation was presented for two different machines, a CRAY YMP and a CRAY C90. All calculations that vectorize are performed in parallel/vector mode. A sustained performance of 724 Mflops on the YMP was observed. Calculation rates of 3.18 Gflops were achieved on the C90. These results were only possible to obtain due to the extensive use of element-by-element iterative techniques.

ACKNOWLEDGEMENTS

The partial support of PETROBRAS S.A., the Brazilian Oil Company, through grant 650.401.792-1 is gratefully acknowledged. This work is part of a joint research project on Finite Elements for Reservoir Simulation between COPPE/UFRJ, LNCC/CNPq and PETROBRAS S.A. Computer time on the CRAY YMP was provided by the National Supercomputing Center of the Federal University of Rio Grande do Sul (CESUP/RS), Brazil. CRAY Research Inc. provides the computational resources on the C90. Special thanks to Ms. D.G. Ewald from CESUP/RS and Mr. T. Bulhões from CRAY Brazil by their invaluable help. We are also indebted to Dr. B. Van Bloomen Waanders, our host in CRAY Research Inc. at Eagan, Minnesota, USA.

REFERENCES

1. Young, L.C. and Hemanth-Kumar, K.: "High Performance Black-Oil Computations", Proc. 11th *SPE* Symposium on Reservoir Simulation, *SPE* 21215, (1991), 135-44.

2. Young, L.C. and Hemanth-Kumar, K.: "Parallel Reservoir Computations", Proc. 13th *SPE* Symposium on Reservoir Simulation, *SPE* 29104, (1995), 101-10.

3. Fung, L. S-K., Hiebert, A.D. and Nghiem, L.X.: "Reservoir Simulation with a Control-Volume Finite Element Method", *SPERE*, (Aug. 1992), 349-57.

4. Loula, A.F.D., Guerreiro, J.N.C., Ribeiro, F.L.B. and Landau, L.: "Tracer Injection Simulations by Finite Element Methods", 3rd Latin American/Caribbean Petroleum Engineering Conference, *SPE* 27047, (1994), 403-410.

5. Tezduyar, T., Aliabadi, S., Behr, M., Johnson, A., and Mittal, S. :"Parallel Finite Element Computation of 3D Flows", *IEE* Computer, October (1993) 27-36.

6. Behr, M., and Tezduyar, T. :"Finite Element Solution Strategies for Large-Scale Flow Simulations", *Comp. Meth. Appl. Mech. Engng.*, (1994), **112**, 3-24.

7. Coutinho, A.L.G.A., Alves, J.L.D., Garcia, E.L.M. and Loula, A.F.D.: "Solution of Miscible and Immiscible Flows Employing Element-by-Element Iterative Strategies", 3rd Latin American/Caribbean Petroleum Engineering Conference, *SPE* 27050, (1994), 431-444.

8. Chavent, G. and Jaffre, J.: *Mathematical Models and Finite Elements for Reservoir Simulation*, Elsevier Science Publishers B.V., Amsterdam, (1986).

9. Brooks, A.N. and Hughes, T.J.R.: "Streamline Upwind/Petrov-Galerkin Formulation for Convection Dominated Flows with Particular Emphasis on the Incompressible Navier-Stokes Equations", *Comp. Meth. Appl. Mech. Engng.*, (1982), **32**, 199-259.

10. Hughes, T.J.R., Mallet, M and Mizukami, A.: "A New Finite Element Formulation for Computational Fluid Dynamics: II. Beyond SUPG", *Comp. Meths. Appl. Mech. Engng.*, (1986), **54**, 341-55.

11. Hughes, T.J.R.: *The Finite Element Method*, Prentice-Hall, Englewood Cliffs, NJ, (1987),803.

12. Gustafsson, K.: "Using Control Theory to Improve Stepsize Selection in Numerical Integration of ODE",Proc. 11th *IFAC* World Congress, Tallinn, Estonia, Vol.1, (1989),139-44.

13. Shakib, F. ,Hughes, T.J.R. and Johan, Z.: "A Multi-Element Group Preconditioned GMRES Algorithm for Nonsymmetric Systems Arising in Finite Element Analysis", *Comp. Meths. Appl. Mech. Engng.*, (1989), **75**, 415-56.

14. Crowl, L.A.: "How to Measure, Present, and Compare Parallel Performance", *IEE Parallel and Distributed Technology.*, Spring (1994), 9-25.

15. Douglas, J., Jr., Wheeler, M.F., Darlow, B.L. and Kendall, R.P.:"Self-Adaptive Finite Element Simulation of Miscible Displacement in Porous Media", *Comp. Meth. Appl. Mech Engng.*, (1984), **47**, 131-59.

16. Abbaszadeh-Dehghani, M. and Brigham, W.E.: "Analysis of Unit Mobility Ratio Well-to-Well Tracer Flow to Determine Reservoir Heterogeneity", Stanford University Technical Report, SUPRI TR-36, Stanford, CA, (1983).

17. Bailey, D.H., Barszcz, E., Dagum, L. and Simon, H.D.: "NAS Parallel Benchmark Results", *IEE Parallel and Distributed Technology.*, February (1993), 43-51.

18. Simon, H.D., Van Dalsem, W.R. and Dagum, L. : "Parallel CFD: Current Status and Future Requirements", *Parallel Computational Fluid Dynamics: Implementation and Results*, Edt. H.D. Simon, MIT Press (1992), 1-32.

APPENDIX

Without loss of generality, the absolute permeability tensor in two-space dimensions has the form:

$$K(x) = k_x \mathbf{i} \otimes \mathbf{i} + k_y \mathbf{j} \otimes \mathbf{j} \tag{A-1}$$

Tensors A and D are given respectively by,

$$A(u,x) = \frac{K(x)}{\mu(u)} \tag{A-2}$$

$$D(v,x) = \left(\frac{\alpha_{mol}}{\tau} + \alpha_T \|v\| \mathbf{I} \right) + \frac{\alpha_L - \alpha_T}{\|v\|} v \otimes v \tag{A-3}$$

where $\mu(u)$ is the viscosity of the fluid mixture, which may be expressed by the following mixture rule,

$$\mu(u) = \left(1 - u + M^{0.25} u\right)^{-4} \mu_r \tag{A-4}$$

The mobility ratio, M, is defined as the ratio of the viscosities of the reservoir fluid (μ_r) and the displacing fluid (μ_d). In Eq.(A-3) the dispersion-diffusion tensor D is expressed in terms of the molecular diffusion (α_{mol}), the longitudinal and transversal hydrodinamic dispersion coefficient (α_L and α_T) and the tortuosity, τ. In tracer flow simulations usually it is assumed a unit mobility ratio, uncoupling pressure and velocities from the concentration equation.