

Large Scale Parallel Reservoir Simulations on a Linux PC-Cluster

Walid A. Habiballah and M. Ehtesham Hayder

Petroleum Engineering Application Services Department
Saudi Aramco, Dhahran 31311, Saudi Arabia
{habibawa@aramco.com.sa and hayderem@aramco.com.sa}

Abstract:

Numerical reservoir simulation is an important tool used by engineers to develop producing strategies for oil and gas fields and to understand fluid flow behavior in subsurface reservoirs. Demand for *mega* scale reservoir simulations is increasing particularly in large fields common in Saudi Arabia. These *mega* cell simulations are expensive on traditional supercomputers. PC cluster technology is rapidly evolving and promises to be a low cost alternative to traditional supercomputers. In this work, we investigated the use of state of the art PC clusters in simulation of massive reservoir models. Our investigation indicates that large scale reservoir simulations can be performed efficiently and cost effectively on latest PC clusters. In this paper, we discuss some of our findings and issues related to parallel reservoir simulations on PC clusters. We also present performance comparison among Xeon Linux PC clusters and an IBM SP Nighthawk supercomputer.

Keywords: Parallel reservoir simulations, PC cluster, Linux cluster, parallel computations.

1. Introduction

POWERS^{1,2} is Saudi Aramco's in-house developed parallel reservoir simulator. This simulator is capable of simulating black-oil, compositional, dual porosity dual permeability and locally refined grid models. Currently POWERS is used to simulate reservoirs models with sizes ranging from few hundred thousands to ten million cells. Over the next few years, number and size of reservoir models are expected to grow

¹ © Saudi Aramco (2003). All rights reserved. No portion of this article may be reproduced by any process or technique without the express written consent of Saudi Aramco. Permission has been granted to QuarterPower Media to publish this work.

significantly. The cost of such simulations on traditional supercomputers will be exorbitantly high. The motivation behind this investigation was to identify and evaluate low cost computing platforms, which will meet increasing computational need of reservoir simulations at Saudi Aramco.

PC-Clusters³ are enjoying growing interest as a high-performance computing platform and the number of installed clusters have been growing (see the Top500 supercomputer list⁴). The PC clusters approach has been proven to be a powerful platform for many large scale scientific computations (see references⁵⁻⁷). Indeed, Saudi Aramco has applied and benefited from use of this technology -platform for its geophysical seismic data processing⁸.

A PC-Cluster is a collection of low cost commodity computers (PCs) interconnected by a switch. Although the notion of building a cluster with commodity hardware is simple, performance or throughput of such a cluster can be quite impressive. This was made possible because the computational engine (the processor) has improved significantly, most notably in last two years. At present, performance of processors in a typical PC cluster (2.4 GHz Pentium IV) is superior to that of a traditional supercomputer. Fig. 1 shows the SPECfp2000 numbers⁹ for three different processors. The Power 3 processor is used in IBM-NightHawk system, the MipsPro processor is used in SGI-Origin systems, while the Pentium IV processor is typical in desktop PCs. While the SPECfp2000 number is only one indicator, the ultimate measure of performance is the performance of the target application.

PC-Clusters have the advantage of price performance ratio over traditional supercomputers. Intel chips are manufactured in the millions compared to processors designed for *special* use. This has made considerable contribution to price reduction of Intel chips and hence PC-Cluster evolution.

At present, possible computing platforms for parallel reservoir simulation are *typical* high-performance computing system (such as IBM-PWR3/4 or SGI-Origin) and PC-Clusters. The cluster platform has the advantage of having high speed processors. For a tightly coupled system such as reservoir simulation, performance of interconnection network is very important. High-speed switches such as Quadrics and Myricom-Myrinet are currently available for building powerful clusters.

IBM SP Nighthawk (NH2) nodes are currently used as the platform for production reservoir simulations at Saudi Aramco. POWERS uses both shared and distributed memory parallel programming models and is fine-tuned to run optimally on IBM Nighthawk nodes. These nodes use 64-bit addressing space and support both shared (OpenMP directive) and distributed memory (MPI calls) programming models. To investigate PC-Clusters, we acquired a 32 node test cluster based on Intel Xeon 2.0 GHz processors connected by Quadrics switch. Later we acquired a system with Mericom-Myrinet switch. Like IBM Nighthawks, our Xeon clusters support both MPI calls and OpenMP directives for parallelization (MPI among all processors, and OpenMP between two processors within a node). Since each node on our clusters has

only two processors, the scope of parallelization using OpenMP directives is limited. Our preliminary experience indicates that the overhead associated with inter-node communication on our current hardware is less than the overhead associated with intra node (i.e., between two processors within a node) parallel computation. This was true for both OpenMP and MPI implementations for parallel computations on two processors within a node.

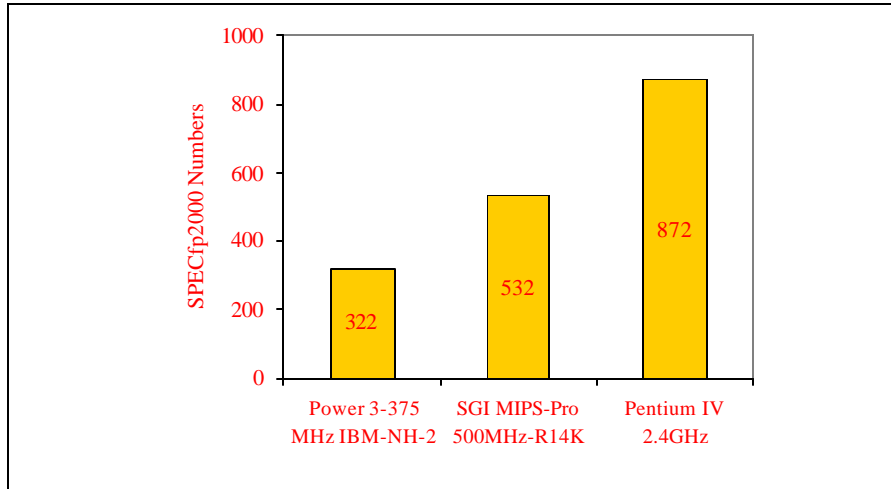


Fig. 1. Performance of different processors

We extended parallelization and revised communication algorithms in POWERS to improve performance on our clusters. We tested our new code on various simulation models. The size of our test models ranged from a few hundred thousand cells to about 10 million cells, numbers of wells were up to 4,000. In this paper we examine issues which are important to obtain good performance on large number processors in a PC-Cluster.

In section 2 we discuss POWERS formulation. Next we describe our hardware platforms in section 3. Our experiences with porting POWERS onto our PC clusters are given in section 4. In section 5, we discuss our experiences with POWERS computations on test platforms. Finally, in section 6, we give our conclusions.

2. Formulation of the Simulator

POWERS uses a finite volume formulation to compute flows in a reservoir and solves mass conservation equations for multi-component multiphase fluid flows in

porous media. Governing equations are derived from mass conservation equations and are written for each components (oil, water and gas) as

$$\frac{\partial(\mathbf{r}_a S_a)}{\partial t} + \nabla \cdot U_a = q \quad (1)$$

where U_a is the Darcy's velocity for each component and is computed as

$$U_a = -\frac{K_a S_a K}{\mathbf{m}} \mathbf{r}_a (\nabla P_a - \mathbf{r}_a g \nabla H) \quad (2)$$

Volume balance equation is used as a constraint given by

$$\sum S_a = 1 \quad (3)$$

If fluid properties are functions of only surrounding pressure and bubble-point pressure, a simplified approach known as black oil formulation is used. In such models, there may be up to three fluid components (water, oil and gas). Some reservoirs are above bubble point pressure and modeled with only two components, namely water and oil. It is important to consider more detailed fluid compositions to determine properties in volatile oil reservoirs and also where enriched gas is injected for enhanced recovery. These reservoirs are modeled using compositional formulation where thermodynamic equilibrium calculations are performed using an equation of state. Reservoir fluid in these models is composed of water and a number of hydrocarbon components. In black oil simulations, solutions are obtained in terms of pressure and saturations by either Implicit Pressure Explicit Saturation (IMPES) or a fully implicit method. Heterogeneous nature of rocks in a fractured reservoir may be modeled using Dual Porosity and Dual Permeability (DPDP) formulation. In DPDP formulation, reservoir rock is represented by a collection of highly permeable fractures and low permeable matrices (or homogeneous solid rocks). Flows in fractures and matrices are coupled. Because of their very small sizes, typical fractures cannot be resolved on a computational grid. Therefore, each grid cell is modeled to contain a lumped fracture and a matrix region.

Rock descriptions in a simulation model are derived from an underlying geological model of the reservoir. Typically, many grid cells of a geological model are consolidated into a single simulation grid cell. This process is known as up scaling. Proprietary software packages are used to upscale geological model data into simulation input data for POWERS. In house software packages are used to generate well production, injection and completion histories directly from Saudi Aramco's corporate database. These data are used for history matching simulation. In a prediction study, computations are guided by production strategies given in simulator input data. Such computations take into account of various parameters such as the production target specified by the user, limitations of surface facilities, well rate/pressure restrictions, etc.

Governing equations describing flows are discretized to get a set of equations for primary variables. These equations are linearized and solved using Newton-Raphson technique. Linearized system of equations is solved within each Newton step using a parallel linear solver known as the reduced system solver. Solution technique used in POWERS is based on generalized conjugate residual (GCR) method and orthomin with truncated Neuman series preconditioner.

POWERS uses three dimensional structured grid cells to discretize the reservoir. In general, flows inside a reservoir can be reasonably represented on a non-uniform structured grid. However, for accurate simulation, one may need high resolution near a well, while such resolution may not be necessary away from a well. In those cases, computational grid near a well can be refined to generate a composite grid consisting of a base grid and locally refined grid (LGR) patches. Current version of POWERS supports two level grid systems, the base grid and one level fine grid patches. An iterative multi grid scheme is used to solve governing equations. At each step of iteration, the coarse grid solutions are obtained and they are used to update residuals on the fine grid. Fine grid equations are then solved and solutions are used to update residuals on the base grid. This iterative process is repeated until a converged solution is obtained.

3. Hardware platforms

Our present study was done on an IBM SP Nighthawk (NH2) and two PC-Cluster systems. IBM NH2 uses 375 MHz Power3 processors and currently is the system used for production reservoir simulations at Saudi Aramco. Our first PC-Cluster uses the Quadrics switch, and the second uses the Mericom-Myrinet switch. The Quadrics cluster was used for initial porting and testing of POWERS. We will refer the Quadrics cluster as Cluster_Q and the Mericom-Myrinet cluster as Cluster_M in this paper. We considered five important items in configuring our PC clusters --- processor, network connection, memory, I/O, and software tools available for development and support of our simulator and simulation activities.

Selection of the cluster processor was fairly straight forward, Pentium IV had advantage over the Pentium III (higher clock rates, better floating point performance, higher bus speed, and the PIV had been just released while PIII had been at the end of its development life). The Xeon gave us the dual-processor capability over the PIV and would allow us to use multi-threading in our code. Rack mounted Xeon clusters were available from numerous vendors and allowed us to have an open bid instead of sole source acquisition.

For the inter-node network connections, we considered four options --- Fast Ethernet, Giga-bit, Mericom-Myrinet, and Quadrics-Elan switches. Specifications of these switches are shown in Table 1.

Switch	Bandwidth (Mbytes/sec)	Latency (microsecond)
Fast Ethernet	12	150
Gigabit	100	26
Myrinet	421	7
Quadrics	400	4

Table 1. Specifications of various switches

High performance parallel reservoir simulation necessitates the use of high-speed communication hardware. The communication between compute nodes is required to compute derivatives at sub domain (grid block) boundaries. Fast Ethernet and Gigabit are low-cost commodity switches with limited bandwidth. They are used for I/O and cluster management but are unsuitable for data communication among grid blocks (at least 200 Mbytes/sec bandwidth is desirable). Mericom-Myrinet and Quadrics-Elan switches have high bandwidth and are more suitable for parallel reservoir simulations.

The system based on Quadrics switch has 32 nodes of dual Intel Xeon processors interconnected with a Quadrics switch. Networks and hardware configurations of this system is shown in Fig. 2. At the time of our hardware acquisition, rack-mounted Xeon-based systems were not available. Therefore, we chose to go with a cluster of workstations based on Xeon systems. Our system consisted of 32 Dell 530 workstations each with two Xeon two GHz processors and four GB memory. One of the 32 nodes was acting as the master node. The master node was connected to the compute nodes via a gigabit switch in addition to the Quadrics network. In addition to doing computations, the master node was also acting as a file server for the compute nodes, a submission and a login node for users. Jobs were submitted from the master node using Resource Management System (RMS) software. Quadrics provides us with RMS to manage parallel tasks across multiple nodes. Every node in this cluster have access to two networks --- the compute network (Quadrics network) for communications in simulations, and the management network (Fast Ethernet) for system administration activities.

The Cluster with Mericom-Myrinet switch consists of two sub-clusters, each of which has 128 compute nodes a master node, a management node and external storage. Each compute node has a Myrinet card and connects to the sub-cluster Myrinet switch. In addition, each compute node is also connected to the rack switch via Fast Ethernet. Each of the compute nodes has two 2.4 GHz Xeon processors and 4 GB memory. The master node is a two-processor server where parallel jobs are initiated. It has ten TB external storage connected to it. Details of our clusters can be found in references 10 and 11.

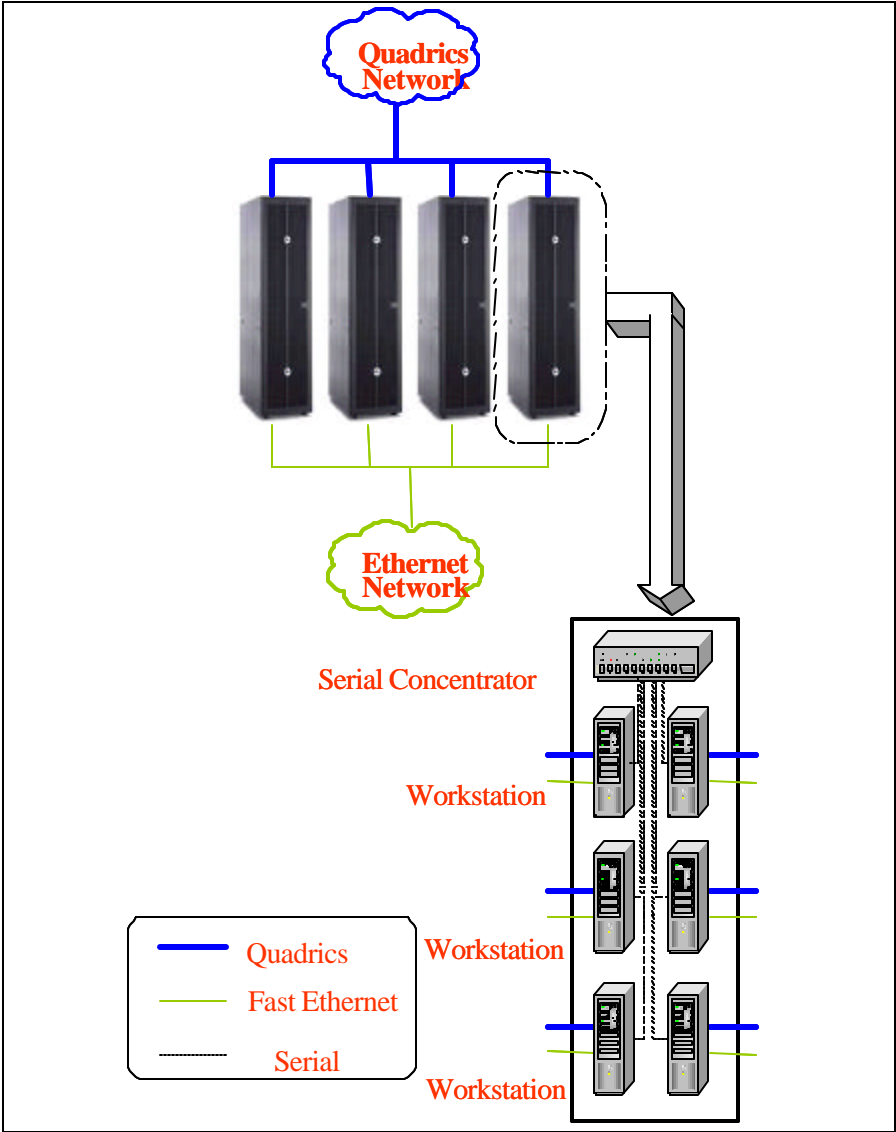


Fig. 2. Configuration of Cluster_M

4. Implementation of POWERS on the PC-Cluster

POWERS, originally written in Connection Machine FORTRAN (CMF), was ported to platform *independent* architecture several years ago. The Parallelization scheme in this code supports both shared and distributed memory hardware architectures. This is done by explicit OpenMP directives and MPI calls. Ideally shared memory parallelism is triggered among multiple processors on the same node via OpenMP directives and distributed memory parallelism across multiple nodes is triggered via MPI calls (MPI can also be used for communication between processors within the same node). POWERS is a dual-parallel code, it uses shared memory programming or OpenMP directive in one spatial direction, and distributed memory programming or MPI calls in another spatial direction. These two parallelization schemes are orthogonal. This orthogonality in parallelization works well in almost all hardware architectures.

We expanded MPI parallelization in a second dimension to take full advantage of PC-Cluster technology. This allowed efficient utilization of the distributed nature of the PC-Cluster architecture, and allowed us to utilize more processors on the cluster. We went through following steps in our implementation and optimization of POWERS on our PC clusters:

- a) Performed straight port (recompiled the code on cluster using the Intel-icc compiler)
- b) Expanded MPI parallelization in second dimension
- c) Benchmarked interconnect switches (Mericom-Merinet and Quadrics-Elan) using code kernels and the full code
- d) Revised and optimized MPI communication algorithm in the code
- e) Tuned performance.

Initially we ported POWERS onto a 32 node Quadrics-cluster (Cluster_Q). We encountered several problems in straight port using the Intel compiler (v.6). POWERS code had extensively used Fortran-95 array syntax with OpenMP directive mixed with MPI calls. Also, the code allocated large chunks of memory (at times reaching two GB per MPI process). This subjected software tools and libraries to significant stress (vendors were very helpful and were able to fix bugs with their software expeditiously). Once POWERS had been functioning correctly on the cluster, the second step was undertaken. Fig. 3 shows an aerial cross-section of the computational grid and parallelization scheme of the original code. The shaded squares represent a collection of cells in the simulation model. The first (I) dimension is used for MPI decomposition among the nodes, the second (J) dimension is used for OpenMP threads. This dual-decomposition divides the computational grid into I-slabs and then every I-slab is divided in the J-direction among threads between the processors within a node.

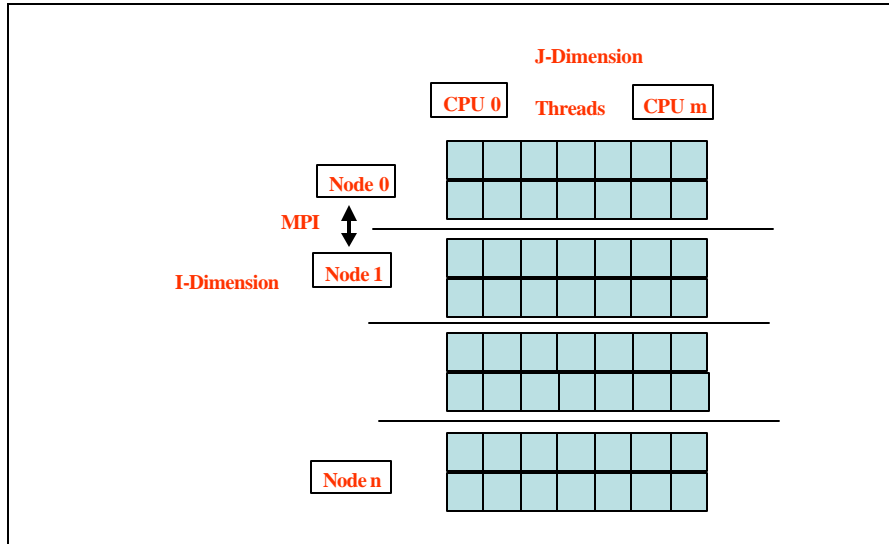


Fig. 3. Parallelization scheme with I-Dimension for MPI, and J-Dimension for OpenMP

Fig. 4 shows our new parallelization scheme. The computational grid is decomposed into aerial blocks and every block is assigned to particular node. Communications (such as those needed for flux calculations) between blocks are done using MPI calls. Furthermore, a node may utilize threads (in J-dimension) for computation within this block. Communication needed for flux calculations depends on the surface area of each decomposed block. Such communications are regular. One can minimize regular communication in flux calculations by choosing a decomposition which minimizes surface area of decomposed domains. There are irregular communications in POWERS for well calculations. Such communications depend on number of decomposed grid blocks (MPI processes), wells and completions. Choice of a particular domain decomposition scheme affects amount of communications and in turn execution time. We briefly examine effects of domain decomposition or blocking in this study. Further discussion of communication issues will be covered in reference 11.

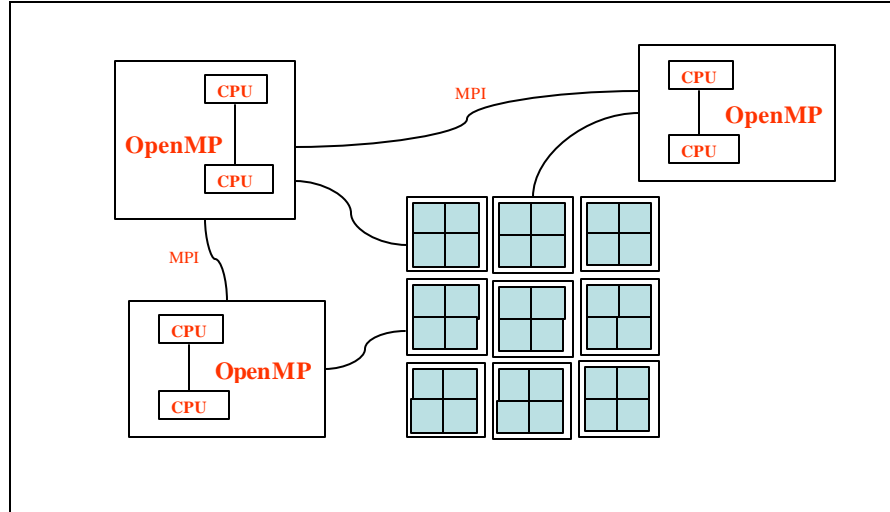


Fig. 4. Parallelization scheme utilizing MPI and OpenMP

5. Results

We evaluated the Quadrics and the Myricom-Myrinet switches as alternative interconnection networks. Our initial results are given in reference 10 and more detailed in reference 11. All computations in this study were done using 64 bit arithmetic. The performance of POWERS on a PC-Cluster depends on a number of factors such as communication to computation ratio, extent of *memory protection* required for multi-threading, size of the problem, amount of I/O, number of wells and completions.

In this paper, we examine some optimization issues related to large scale reservoir simulation on PC clusters. We have utilized four reservoir models (Table 2) for this paper. These four models were selected to examine different computational and communicational characteristics in reservoir simulations. The first two models are large, Model I is a 3.7 million cell model with 144 wells, and Model II is a 9.6 million cell model with about 3000 wells (this is one of the larger reservoir models built in Saudi Aramco). We chose this model to identify and study issues which could be important for large scale simulation on cluster environments. Models III and IV have relatively small number of grid cells. Large numbers of wells in Model III makes it suitable for studying effects of well related computations and communications. Typical simulation result (saturation plot) of Model II is shown in Appendix A.

Model	Grid Size (million cells)	Number of wells	Years Simulated
Model I	3.7	144	3
Model II	9.6	2,960	5
Model III	0.2	4,000	62
Model IV	0.5	200	24

Table 2. Test Cases

We present timings for Model I in Table 3. Aggressive compilation flag (-O3) and other tuning options were used on both IBM NH2 and PC clusters. POWERS was compiled with the Intel compiler (v.6) on both Cluster_Q (cluster with Quadrics switch) and Cluster_M (cluster with Myrinet switch). Performances of Quadrics and Myrinet clusters were within 20% of each other. PC Clusters were about two to three times faster than IBM-NH2 for simulations on same number of processors. Processor speed played an important role in reducing execution times on the clusters. Cluster_Q has 2 GHz and Cluster_M has 2.4 GHz processors compared to 375 MHz Power 3 processors on IBM-NH2. Later Intel compiler on Cluster_M was upgraded from v.6 to v.7. Mericom-Myrinet GM library was also upgraded at the same time when v.7 compiler was installed on Cluster_M. As shown in Fig. 5, these two upgrades reduced execution time by about 8% for 16 processors run and about 25% for 32 processor run.

CPU	Cluster_Q (sec)	Cluster_M (sec)	IBM-NH2 (sec)
16	3,900	3,200	8,960
32	1,960	2,175	4,790

Table 3. Model I Execution time on PC Clusters and IBM_NH2

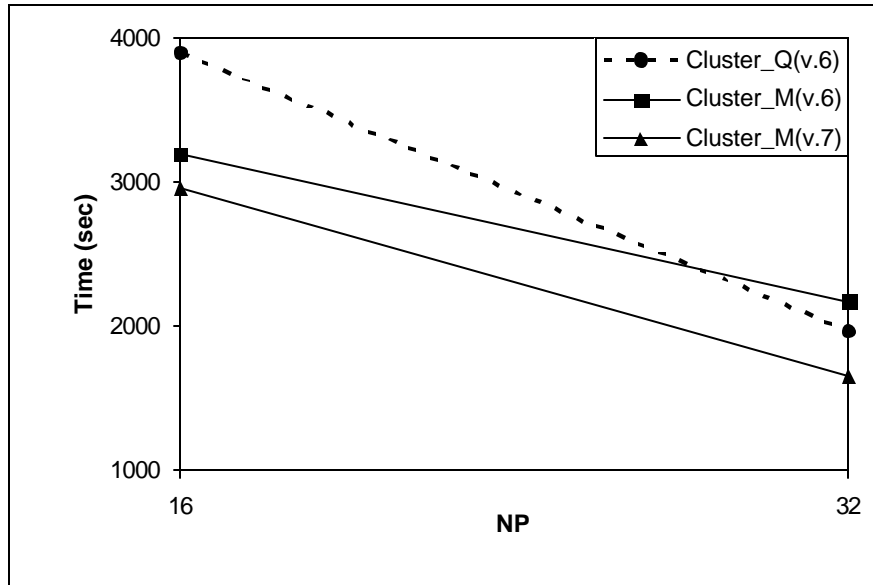


Fig. 5. Execution time of Model I on PC Clusters

We next examine timings and scalability of Model I. Our results on Cluster_M and IBM-NH2 are shown in Table 4 and Fig. 6. We present execution time (T_t) and core simulation time (T_c), which excludes initialization and I/O overheads, in Table 4, and speedups in execution time and speedup in core simulation time in Fig. 6. Simulations were made on up to 120 processors. Speedups were comparable and generally slightly higher on the IBM-NH2 compared to those on the Cluster_M. On 96 processors, speedup on Cluster_M was slightly better than that on IBM-NH2. However, simulation time on the Cluster_M increased from 740 seconds on 96 processors to 770 seconds on 120 processors, although core simulation time decreased from 630 seconds to 520 seconds. Overhead in these two simulations jumped from about 15% on 96 processors to over 30% on 120 processors. We examine overheads on large number processor later in this study.

CPU	Tt (cluster) (sec.)	Tc (cluster) (sec.)	Tt (NH2) (sec.)	Tc (NH2) (sec.)
16	2,960	2830	8,960	8,850
32	1,650	1525	4,790	4,650
48	1,140	975	3,340	3,180
64	960	835	2,650	2,485
80	810	655	2,440	2,265
96	740	630	2,245	2,050
120	770	520	1,785	1,595

Table 4. Model I Simulation time on Cluster_M

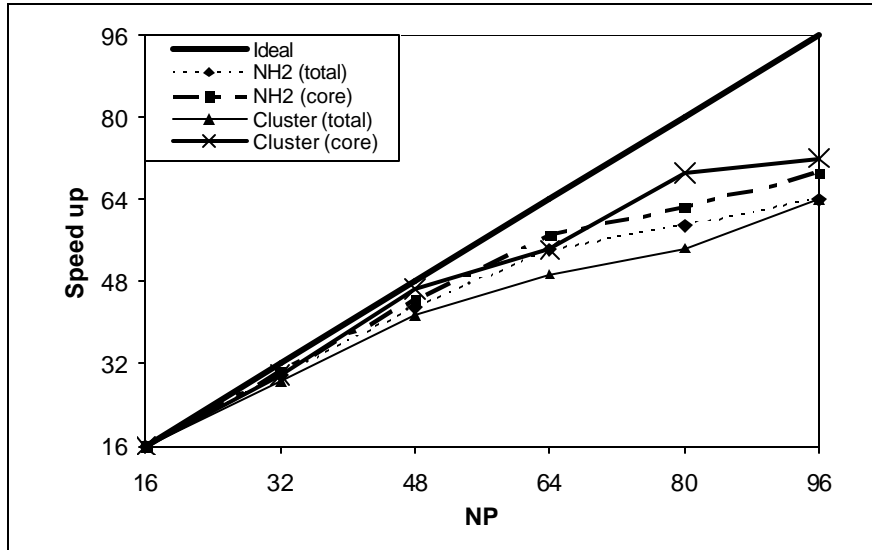


Fig. 6. Scalability of Model I

Next we consider Model II and show our results in Table 5 on up to 120 processors. On smaller number of processors, Cluster_M was much faster than IBM-

NH2. But overheads on the Cluster were high on large number of processors. In fact, on 120 processors core simulation time was about 45% of execution time. To understand the cause for large overheads we examine timings of various sections of the code for simulations of Model I and II. In Figures 7 and 8, we show communication and synchronization overhead times along with execution time (Tt) and core simulation time (Tc) for these model as measured on the master node (node 0).

CPU	Tt (cluster) (sec.)	Tc(cluster) (sec.)	Tt (NH2) (sec.)	Tc (NH2) (sec.)
48	4,300	3,780	11,115	10,890
64	3,590	2,815	8,960	8,735
96	4,140	2,284	6,000	5,700
120	4,190	1,864	4,415	4,180

Table 5. Simulation times of Model II on Cluster_M and IBM NH2

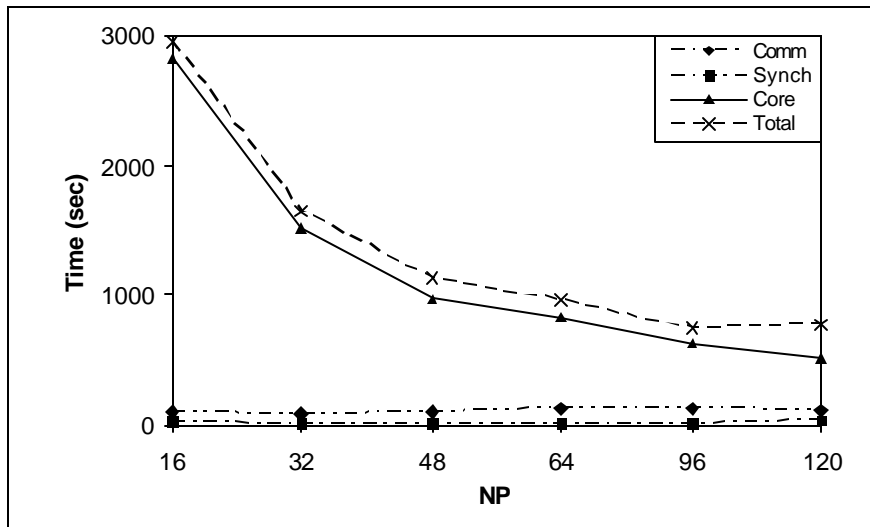


Fig. 7. Timing of Code Sections in Model I

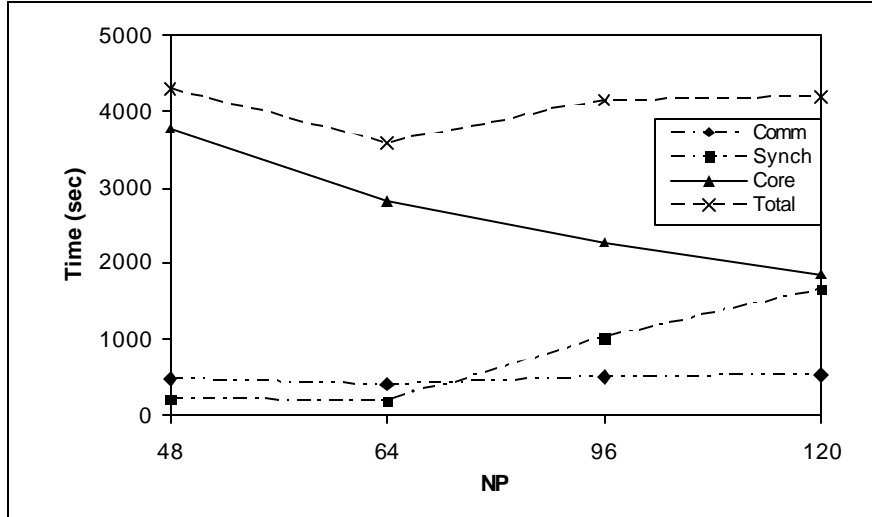


Fig. 8. Timing of Code Sections in Model II

Communication overheads of Model I did not change significantly as number of processors increased. Same was true also for Model II. On 120 processors, communication overhead was about 15% of total execution time for Model I and about 13% of total execution time for Model II. In absence of any other overhead, speedup on 120 processors would be about 100. Synchronization overheads for Model II was very high. This may have been caused by load imbalance in our simulation. We are now trying to reduce overheads by adjusting grid distribution of computing nodes. We are also examining our communication (MPI) routines and trying to reduce unnecessary barriers or blockings in data sends and receives.

Next we present results with an optimized irregular communication algorithm in the code. As indicated earlier, there are two main types of communication in POWERS --- communications for calculation of spatial gradients or fluxes, and communication for mapping well data onto the spatial grid. The former is regular neighbor communication and later is irregular communication. Irregular communication becomes very important when a large number of wells is present in the reservoir. We show execution time and speedups for Model III in Table 6 and Fig. 9 respectively. We also show results on the Cluster_Q (using Intel compiler v.6) and IBM-NH2 for reference. On 20 Processors, our simulation took 8,000 seconds on the Cluster_Q and 7,700 seconds on IBM-NH2. On Cluster_M (using Intel compiler v.7) this simulation took 5700 seconds for the original code and 4,860 seconds for the optimized code. Speedup of this model on IBM NH2 was better than those on clusters. Simulations on NH2 were done with large number of OpenMP processes and small number of MPI processes to keep amount of irregular communication low. On

clusters we had to use large number of MPI processes. In Fig. 10 we show results of Model I T_{opt}/T_{orig} is the ratio of timings with the optimized communication algorithm to those with the original algorithm. Execution time reduced by about 25% on 120 processors when our optimized irregular communication algorithm was used. Timings of regular and irregular communications for Model III are given in Fig. 11. Improvements in communication timings with our revised code are noticeable.

CPU	Cluster_Q (sec.)	Cluster_M(orig) (sec.)	Cluster_M(opt) (sec.)	IBM NH2 (sec.)
1	56,100	45,830	45,830	98,350
10	9,900	7,350	7,090	13,500
20	8,000	5,700	4,860	7,700
50	---	5,230	3,720	4,675

Table 6. Model III execution time

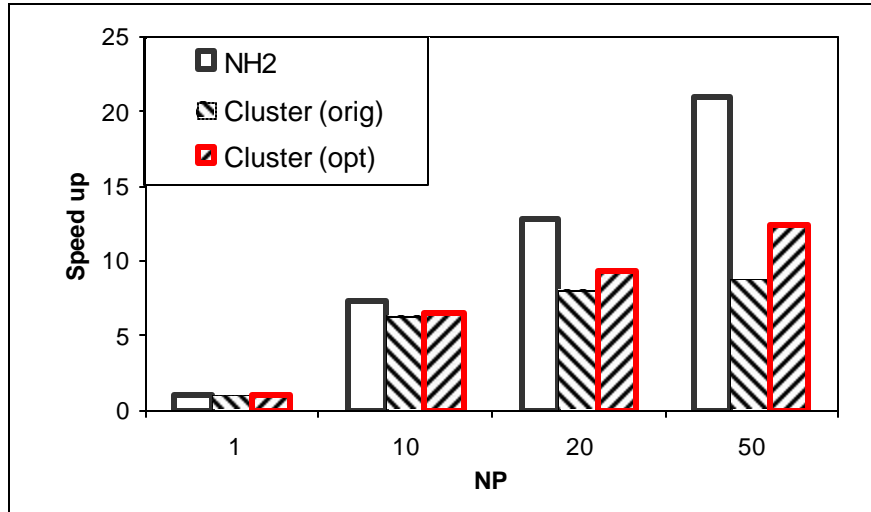


Fig. 9. Scalability of Model III

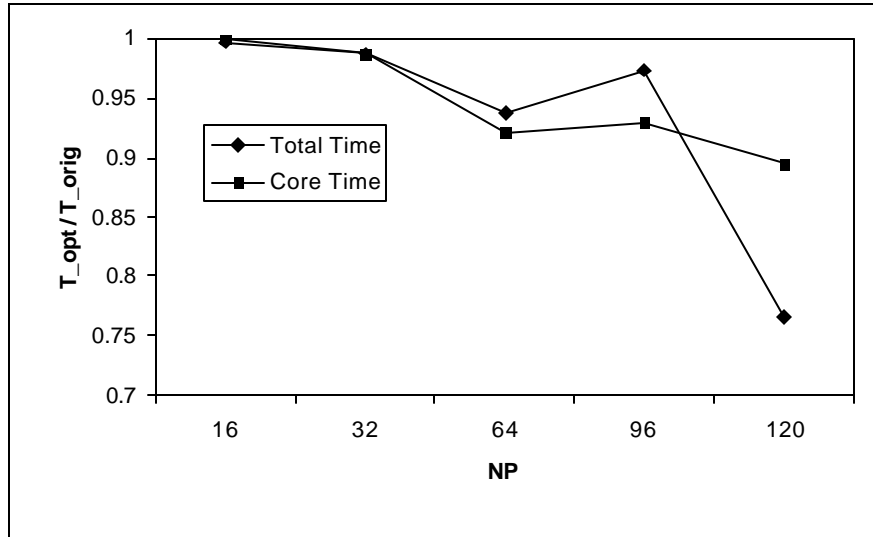


Fig. 10. Effect of optimized communication algorithm for Model I Simulation

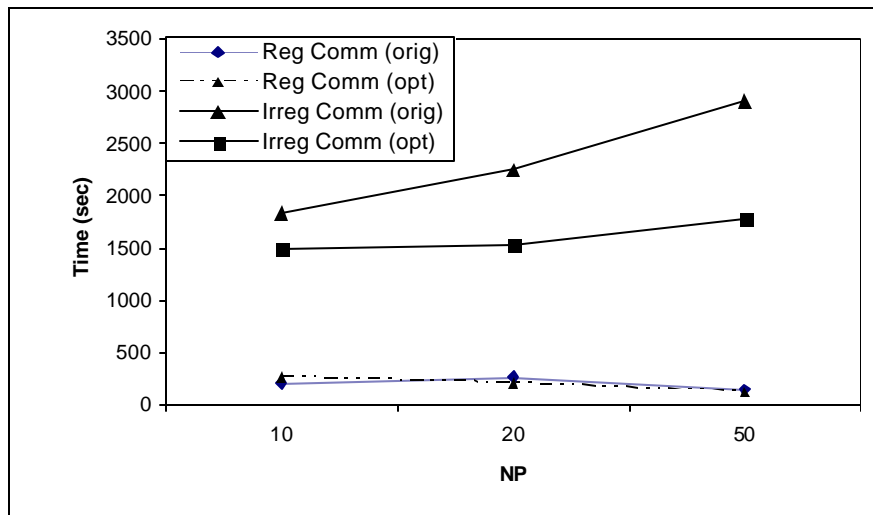


Fig. 11. Communication Overhead in Model III Simulation

We next examine domain decomposition parameters. Results for Model I and IV are given in Fig. 12. Simulations were made on 32 processors. 1x32 blocking (i.e. I dimension is not decomposed and J dimension is decomposed in 32 parts in Fig. 4) was the best for Model I, and 4x8 blocking was best for Model IV. The original code had only I dimension decomposition using MPI processes (Fig. 3), which corresponds to 32x1 blocking in Fig. 12. Optimal blocking parameter depends on the reservoir model (grid parameters).

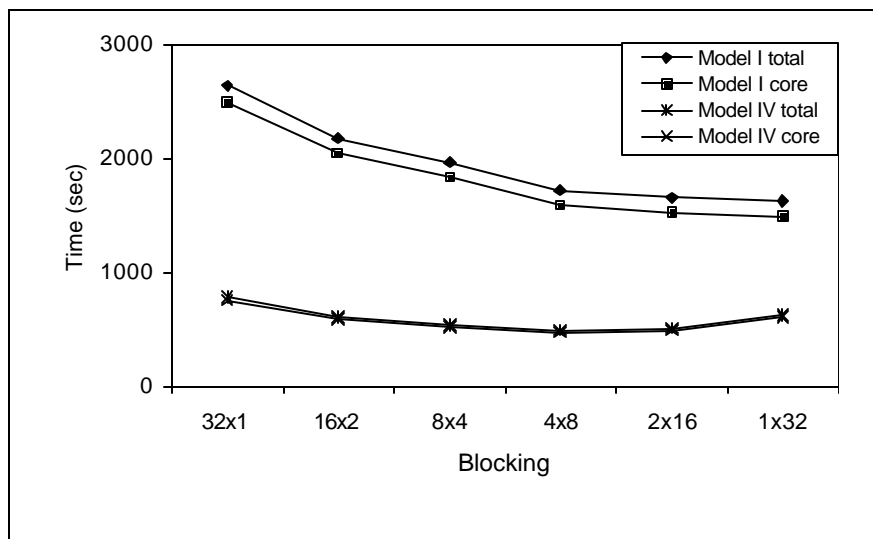


Fig. 12. Timing with Different Decomposition in Models I and IV

6. Conclusions

Based on our observations in this work, we conclude the followings:

1. PC-Clusters are highly attractive option for parallel reservoir simulations in terms of performance and cost.

2. Simulations of massive reservoir models (in the order of 10 million cells) are possible at affordable computing cost also feasible within very short period.
3. Serial computation, synchronization and I/O issues become significant at high number of processors (i.e. 100+ CPUs).
4. Commodity switches can be effectively used in high performance computing platform for reservoir simulation.
5. Both hardware and software tools such as compilers are constantly improving which in turn will reduce execution times and computing costs on PC-clusters.

Acknowledgements

Authors would like to thank Saudi Aramco management for permission to publish this paper. Authors would also like to express their sincere gratitude to N. Al-Rabeh, ECC General Manager for his continuous encouragement and support. They also would like to extend their appreciations to PEASD management, specially to R. Al-Rabeh and A. AlMossa, and to all members of ECC Linux Cluster team members.

Nomenclature

- g = gravitational constant
- H = depth
- K_a = relative permeability of phase a
- K = absolute permeability tensor
- P = Pressure
- q = well flow rate
- S_a = Saturation of phase a
- t = time
- U = Darcy's velocity
- ϕ = porosity
- ρ = density
- μ_a = phase viscosity

References

1. Dogru, A. H., Li, K. G., Sunaidi, H. A., Habiballah, W. A., Fung, L., Al_Zamil, N., and Shin, D.; "A Massively Parallel Reservoir Simulator for Large Scale Reservoir Simulation", SPE paper 51886, Presented at the 1999 SPE Reservoir Simulation Symposium, February 1999.
2. Fung, L. S. and Dogru, A. H.: "Efficient Multilevel Method for Local Grid Refinement for Massively Parallel Reservoir Simulation", *Paper M-28, Presented at the ECMOR VII – European Conference on the Mathematics of Oil Recovery*, Lago Maggiore, Italy, September 2000.
3. Buyya, R. and Baker, M., "Cluster Computing at a Glance", *High Performance Cluster Computing* Vol. 1, pp 3-45, Prentice-Hall Inc., NJ, 1999.
4. <http://www.top500.org>
5. Hayder, M. E. and Jayasimha, D. N.: "Navier-Stokes Simulations of Jet Flows on a Network of Workstations", *AIAA Journal*, 34(4), pp 744-749, 1996.
6. Bhandankar, S. M. and Machaka, S. "Chromosome Reconstruction from Physical Maps Using Cluster of Workstations", *Journal of Supercomputing*, 11(1), pp 61-86, 1997.
7. Komatitsch, D. and Tromp, J.: "Modeling of Seismic Wave Propagation at the Scale of the Earth on a Large Beowulf", *Presented in the SC2001* (<http://www.sc2001.org>), Nov 2001.
8. Huwaidi, M. H., Tyraskis, P. T., Khan, M. S., Luo, Y. and Faraj, S. A.: "PC-Clustering at Saudi Aramco: from Concept to Reality", *Saudi Aramco Journal of Technology*, Spring 2003, pp 32-42. 2003.
9. <http://www.spec.org>
10. Habiballah, W., Hayder, M., Uwaiyedh, A., Khan, M., Issa, K., Zahrani, S., Tyraskis, T., Shaikh, R., and Baddourah, M.: "Prospects of Large Scale Reservoir Simulations at Saudi Aramco on PC Clusters", *Presented at the 2003 SPE Technical Symposium of Saudi Arabia*, Dhahran, Saudi Arabia, June 2003.
11. Habiballah, W., Hayder, M., Uwaiyedh, A., Khan, M., Issa, K., Zahrani, S., Tyraskis, T., Shaikh, R., and Baddourah, M.: "Parallel Reservoir Simulation Utilizing PC-Clusters in Massive Reservoirs Simulation Models", *To be presented at the SPE Annual Technical Conference and Exhibition*, Denver, CO, USA. October 2003.

Appendix A

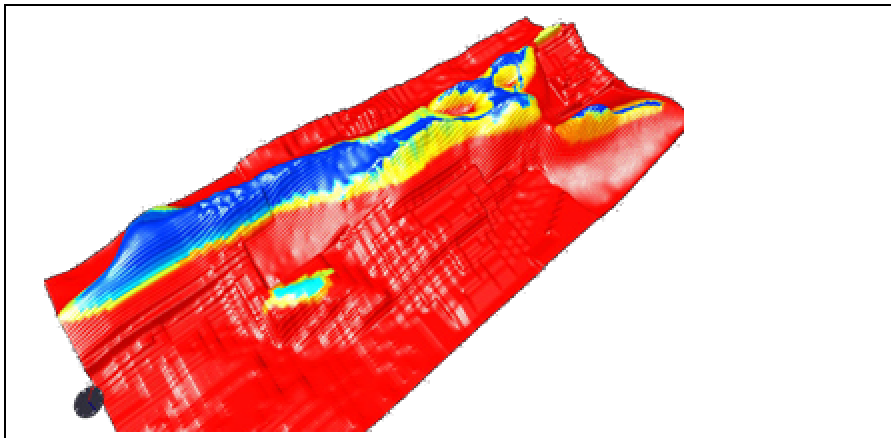


Fig. 13. A Saturation Plot of Model II