Application of Grid-Enabled Technologies for Solving Optimization Problems in Data-Driven Reservoir Studies*

Manish Parashar[1], Hector Klie[2], Umit Catalyurek[3], Tahsin Kurc[3], Vincent Matossian[1], Joel Saltz[3], and Mary F. Wheeler[2]

[1] Dept. of Electrical & Computer Engineering,
Rutgers, The State University of New Jersey, New Jersey, USA
{parashar,vincentm}@caip.rutgers.edu
[2] CSM, ICES, The University of Texas at Austin, Texas, USA
{klie,mfw}@ices.utexas.edu
[3] Dept. of Biomedical Informatics, The Ohio State University, Ohio, USA
{umit,kurc,jsaltz}@bmi.osu.edu

Abstract. This paper presents use of numerical simulations coupled with optimization techniques in reservoir modeling and production optimization. We describe three main components of an autonomic oil production management framework. This framework implements a dynamic, data-driven approach and enables execution in a Grid environment for large scale optimization formulations in reservoir modeling.

1 Introduction

The ultimate goal of reservoir simulation is to generate both good estimates of reservoir parameters and reliable predictions of oil production to optimize return on investment from a given reservoir. The objective function $f(w, s)$ can be seen in terms of a performance measure depending on a vector of decision variables w (e.g., well location indices) and on a vector of uncontrollable conditions s (e.g.,rock porosity values, oil and water saturation). Function f usually represents a mismatch between observed and computed values (history matching) or a economical model based on the amount of oil produced or displaced. In either case, since sampling locations in the field are sparse and the amount of information is scarce, solutions to f are plagued with sources of error and uncertainity[1]. This means that any objective function is limited to a partial knowledge of reality.

* This work is partly supported under the National Science Foundation under Grants ACI-9619020 (UC Subcontract 10152408), ANI-0330612, EIA-0121177, SBR-9873326, EIA-0121523, ACI-0203846, ACI-0130437, ACI-9982087, NPACI 10181410, ACI 9984357, EIA 0103674 and EIA- 0120934, Lawrence Livermore National Laboratory under Grant B517095 (UC Subcontract 10184497), Ohio Board of Regents BRTTC BRTT02-0003, and DOE DE-FG03-99ER2537.

[1] Note, that in order to simplify the discussion we have omitted the use of constrains which are also common in this setting: fixed budget, allowable locations, predefined surface facilities, to name a few.

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3038, pp. 805–812, 2004.

Black oil and more complex compositional, geomechanical, thermal, and chemical models can be used as forecasting tools in both day-to-day operational management of production facilities and long term field development planning. However, little use has been made of reservoir simulations coupled with systematic optimization techniques. The main advantage of applying these mathematical tools to decision-making process is that they are less restricted by human imagination than conventional case-by-case comparisons. A key issue is to come up with reliable prediction models, despite the inherent uncertainty and scales involve in all subsurface measurements, that operate by searching a large space of oil production and reservoir parameters.

One of the main obstacles to the application of optimization techniques coupled with a reservoir simulator is the computational time required to complete simulations of complex, large scale reservoir models. Optimization strategies normally evaluate hundreds or even thousands of scenarios (each representing a simulation run) in the course of searching for the optimal solution to a given management question. This process is extremely time-consuming and data-intensive [5, 8] and can easily overwhelm local computational capacity at any single institution. This approach is further hampered by the need to navigate multi-terabyte datasets from simulations and field measurements.

Grid computing is rapidly emerging as the dominant paradigm for large-scale parallel and distributed computing. A key contribution of Grid computing is the potential for seamless aggregations of and interactions among computing, data and information resources, which is enabling a new generation of scientific and engineering applications that are self-optimizing and dynamic data driven. However, achieving this goal requires a service-oriented Grid infrastructure that leverages standardized protocols and services in accessing hardware, software, and information resources [4,8].

In a previous work, we described a suite of tools and middleware that enable analysis of large, distributed collections of simulation datasets [13]. In this paper, we present an infrastructure for solving optimization problems in data-driven reservoir simulations in the Grid. The infrastructure builds on 3 key components; a computational engine consisting of a simulation framework (IPARS) and optimization services, middleware for distributed data querying and subsetting (STORM), and an autonomic Grid middleware (Discover) for service composition, execution, and collaboration. We describe each of these components and their application in autonomic data-driven management of the oil production process [9].

2 Computational Grid Components

2.1 The Integrated Parallel Accurate Reservoir Simulator (IPARS)

IPARS represents a new approach to a parallel reservoir simulator development, emphasizing modularity of code portability to many platforms, and ease of integration with other software. It provides a set of computational features such as memory management for general geometric grids, portable parallel communication, state-of-the-art non-linear and linear solvers, keyword input and output

for visualization. There are currently several models in IPARS, including multiphase gas-oil-water, air-water and one-phase flow, compositional, geomechanical and reactive transport models. The framework supports both the use of IMPES (implicit pressure explicit saturations) and fully implicit formulations. A key feature of IPARS is that it allows the definition of different numerical and physical models in different blocks of the domain (i.e., multi-numeric, multiphysics and multiblock capabilities). A more technical description of IPARS with further applications can be found in [1].

2.2 Optimization Algorithms

Very Fast Simulated Annealing(VFSA). This algorithm is a simulated annealing variant the speedups the process by allowing a larger sampling at the beginning and a much narrower sampling at its latest stages. This is achieved by the use of a Cauchy like distribution. The second appealing feature is that each model parameter can have its own cooling schedule and model space sampling schemes. This allows selective control of the parameters and the use of *a priori* information (e.g., [14])

Simultaneous Perturbation Stochastic Algorithm (SPSA). The novelty of the SPSA is the underlying derivative approximation that requires only two (for the gradient) or four (for the Hessian matrix) evaluations of the loss function regardless of the dimension of the optimization problem. In other words, it does not require full gradient function information. This feature allows for a significant decrease in the cost of optimization, specially in problems with a large number of decision parameters to be inverted. This algorithm is suitable for noisy measurements of the objective function and the search for a global optimizer (e.g., [15]).

Gradient based. These methods essentially use the approximated gradient of the response surface to derive a search direction. Along the search direction a better point is located based on the response values. Different ways for generating the search direction result in different methods. Newton and quasi-Newton methods[3] and finite-difference stochastic approximation (FESA) methods [15] are representative examples.

Hybrid approaches. These methods are based on the coupling of either the VFSA or the SPSA methods with any of the gradient based methods. This allows to improve the overall convergence of the optimization procedure in the vicinity of the desired solution.

3 Querying and Subsetting of Distributed Data: STORM

STORM (a.k.a. GridDB-Lite) [11] is a services-oriented middleware that is designed to provide basic database support for 1) *selection of the data of interest*: The data of interest is selected based on attribute values or ranges of values, and can involve user-defined filtering operations. 2) *transfer of data from storage nodes to compute nodes for processing*: After the data of interest has been

selected, it can be transferred from storage systems to processor memories for processing by a potentially parallel data analysis program.

STORM supports data select and data transfer operations on scientific datasets through an object-relational database model. With an object-relational view of scientific datasets, the data access structure of an application can be thought of as a *SELECT* operation as shown in Figure 1. The $< Expression >$ statement can contain operations on ranges of values and joins between two or more datasets. *Filter* allows implementation of user-defined operations that are difficult to express with simple comparison operations.

Datasets generated in scientific applications are usually stored as a set of flat files. STORM services provide support to create a view of data files in the form of virtual tables using application specific *extraction* objects. An extraction object is implemented by an application developer and returns an ordered list of attribute values for a data element in the dataset, thus effectively creating a virtual table. The analysis program can be a data parallel program. The distribution of tuples in a parallel program can be represented as a *distributed array*, where each array entry stores a tuple. This abstraction is incorporated into our model by the *GROUP-BY-PROCESSOR* operation in the query formulation. *ComputeAttribute* is another user-defined function that generates the attribute value on which the selected tuples are grouped together based on the application specific partitioning of tuples.

```
SELECT < Attributes >
    FROM Dataset_1, Dataset_2, ..., Dataset_n
    WHERE < Expression > AND Filter(< Attributes >)
    GROUP-BY-PROCESSOR ComputeAttribute(< Attributes >)
```

Fig. 1. Formulation of data retrieval steps as an object-relational database query.

STORM has been developed using a component-based framework, called DataCutter [2], which enables execution of application data processing components in a distributed environment. Using the DataCutter runtime system, STORM implements several optimizations to reduce the execution time of queries: **Distributed Execution of Filtering Operations.** Both data and task parallelism can be employed to execute user-defined filtering operations in a distributed manner. If a select expression contains multiple user-defined filters, a network of filters can be formed and executed on a distributed collection of machines. **Parallel Data Transfer.** Data is transferred from multiple data sources to multiple destination processors by STORM data mover components. Data movers can be instantiated on multiple storage units and destination processors to achieve parallelism during data transfer.

4 An Autonomic Grid Middleware for Oil Reservoir Optimization

Discover [7] enables a seamless access to and peer-to-peer integration of applications, services and resources on the Grid. The middleware substrate inte-

grates Discover collaboratory services with the Grid services provided by the Globus Toolkit using the CORBA Commodity Grid (CORBACoG) Kit [12]. It also integrates the Pawn peer-to-peer messaging substrate [9]. Pawn enables decentralized (peer) services and applications to interact and coordinate over wide area networks. Finally, Discover/DIOS [10] distributed object infrastructure that enables development and management of interaction objects and applications, encapsulating sensors and actuators, and a hierarchical control network. DIOS also allows the dynamic definition and deployment of policies and rules to monitor and control the behavior applications and/or application services in an autonomic manner [6]. Detailed descriptions of the design, implementation and evaluation of Discover components can be found in [7,9,10,6].

5 Putting It Together: Data-Driven Oil Production Optimization

Oil production optimization process involves (1) the use of an integrated multi-physics/multi-block reservoir model and several numerical optimization algorithms (global, local and hybrid approaches) executed on distributed computing systems in the Grid; (2) distributed data archives that store historical, experimental (e.g., data from sensors embedded in the field) and observed data; (3) Grid services that provide secure and coordinated access to the resources required by the simulations; (4) external services that provide data relevant to optimization of oil production or of the economic profit such as current oil market prices, and (5) the actions of scientists, engineers and other experts, in the field, the laboratory, and in management offices.

In this process, item 1 is implemented by the IPARS framework. Both forward modeling (comparison of the performance of different reservoir geostatistical parameter scenarios) and inverse modeling (searching for the optimal decision parameters) for solving optimization problems in reservoir management can greatly benefit from integration and analysis of simulation, historical, and experimental data (item 2). Common analysis scenarios in optimization problems in reservoir simulations involve economic model assessment as well as technical evaluation of changing reservoir properties (e.g., amount of bypassed oil, concentration of oil and water) [13]. In a Grid environment, data analysis programs need to access data subsets on distributed storage systems. This need is addressed by STORM. An example query for exploring regions of bypassed oil in one or more simulation datasets is given in Figure 2. The Discover autonomic Grid middleware implements the support for items 3, 4, and 5. We now discuss the use of Discover to enable oil reservoir optimization [8].

The overall application scenario is illustrated in Figure 3. The peer components involved include: IPARS providing sophisticated simulation components that encapsulate complex mathematical models of the physical interaction in the subsurface, and execute on distributed computing systems on the Grid; IPARS Factory responsible for configuring IPARS simulations, executing them on resources on the Grid and managing their execution; Optimization Service (e.g. very fast simulated annealing); and Economic Modeling Service that uses IPARS

SELECT $R.Cell_x, R.Cell_y, R.Cell_z, R.Id, R.Time$
FROM $Realization_1, Realization_2, ..., Realization_n$
WHERE $T_{start} <= R.Time$ AND $R.Time <= T_{end}$
AND $R.SOIL > SOIL_{tol}$
AND $Speed(R.V_{oil,x}, R.V_{oil,y}, R.V_{oil,z}) < Speed_{tol}$
GROUP-BY-PROCESSOR $Partition(R.Id, R.Time)$

Fig. 2. An example query for analysis of data in oil reservoir management studies: "Retrieve all the mesh cells, from simulations $Realization_1, ..., Realization_n$, which contain bypassed oil (which is defined as cells, in which oil saturation is greater than user-defined oil saturation threshold, $SOIL_{tol}$, and oil speed is less than user-defined speed threshold."

simulation outputs and current market parameters (oil prices, costs, etc.) to compute estimated revenues for a particular reservoir configuration.

These entities need to dynamically discover and interact with one another as peers to achieve the overall application objectives. Figure 3 illustrates the key interactions involved: (1) the experts use the portals to interact with the Discover middle-ware and the Globus Grid services to discover and allocate appropriate resource, and to deploy the IPARS Factory, Optimization Service and Economic model peers. (2) The IPARS Factory discovers and interacts with the Optimization Service peer to configure and initialize it. (3) The experts interact with the IPARS Factory and Optimization Service to define application configuration parameters. (4) The IPARS Factory then interacts with the Discover middle-ware to discover and allocate resources and to configure and execute IPARS simulations. (5) The IPARS simulation now interacts with the economic model to determine current revenues, and discovers and interacts with the Optimization Service when it needs optimization. (6) The Optimization Service provides IPARS Factory with optimized well information, which then (7) launches new IPARS simulations with update parameters. (8) Experts can at anytime discover, collaboratively monitor and interactively steer IPARS simulations, configure the other services and drive the scientific discovery process. Once the optimal well parameters are determined, the IPARS Factory configures and deploys a production IPARS run.

Figure 4 shows the convergence history for the optimization of well location using the VFSA optimization service, to maximize profits for a given economical revenue objective function. The well positions plot (on the left) shows the oil field and the positions of the wells. The black circles represent fixed injeciton wells and the well at the bottom most part of the plot is a fixed production well. The plot also shows the sequence of well position guesses for the other production well returned by the VFSA service (shown by the lines connecting the light squares), and the corresponding normalized cost value (plot on the left).

6 Conclusions

In this paper, we presented an infrastructure and its components to support autonomic oil production management process. Use of this infrastructure to im-

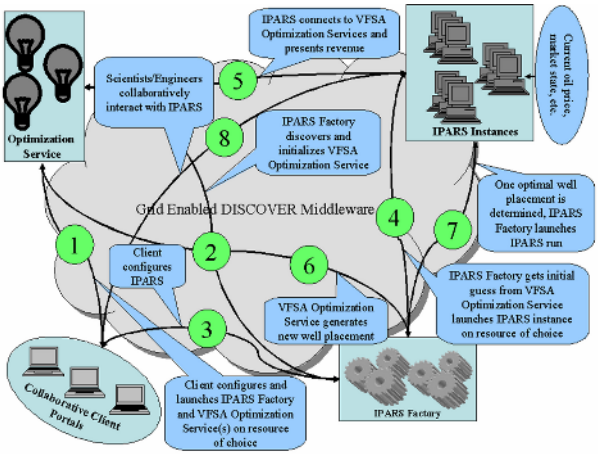

Fig. 3. Autonomous oil reservoir optimization using decentralized services.

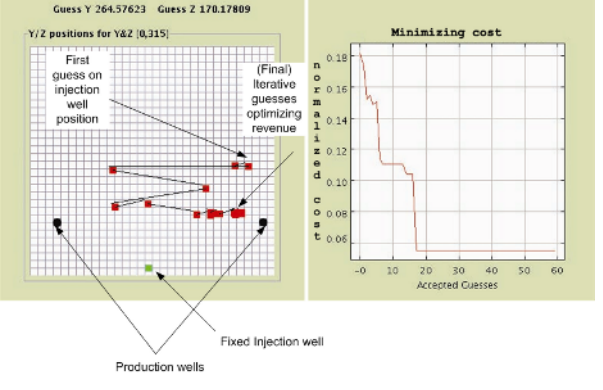

Fig. 4. Convergence history for the optimal well placement in the Grid.

plement Grid-enabled data-driven application support can aid in gaining better understanding of subsurface properties and decision variables. With a better understanding of these properties and variables, engineers and geoscientists can implement optimized oil production scenarios. We believe autonomic oil production management strategies combined with Grid-enabled data and parameter space exploration technologies can lower infrastructure costs and change the economics of productivity maximization.

Acknowledgment. We would like to thank Ryan Martino and Małgorzata Peszyńska for their help in executing the VFSA-based experiments.

References

1. IPARS: Integrated Parallel Reservoir Simulator. The University of Texas at Austin, http://www.ices.utexas.edu/CSM.
2. M. D. Beynon, T. Kurc, U. Catalyurek, C. Chang, A. Sussman, and J. Saltz. Distributed processing of very large datasets with DataCutter. *Parallel Computing*, 27(11):1457–1478, Oct 2001.
3. J. Dennnis and R. Schnabel. *Numerical Methods for Unconstrained Minimization and nonlinear equations*. Prentice Hall, Englewood Cliffs, New York, 1983. reprinted by SIAM, Classics in Applied Mathematics, 1996.
4. I. Foster and C. Kesselman. *The Grid: Blueprint for a new computing infrastructure*. Morgan Kaufmann, Los Altos, CA, 1999.
5. V. M. Johnson and L. L. Rogers. Applying soft computing methods to improve the computational tractability of a subsurface simulation-optimization problem. *Journal of Petroleum Science and Engineering*, 29:153–175, 2001.
6. H. Liu and M. Parashar. Dios++: A framework for rule-based autonomic management of distributed scientific applications. In H. Kosch, L. Boszormenyi, and H. Hellwagner, editors, *Proceedings of the 9th International Euro-Par Conference (Euro-Par 2003)*, volume 2790 of *Lecture Notes in Computer Science*, pages 66–73. Springer-Verlag, August 2003.
7. V. Mann and M. Parashar. Engineering an Interoperable Computational Collaboratory on the Grid. *Grid Computing Environments. Special Issue of Concurrency and Computations: Practice and Experience*, 14(13-15):1569 – 1593, 2002.
8. V. Matossian and M. Parashar. Autonomic optimization of an oil reservoir using decentralized services. In *Proceedings of the 1st International Workshop on Heterogeneous and Adaptive Computing– Challenges for Large Applications in Distributed Environments (CLADE 2003)*, pages 2–9. Computer Society Press, June 2003.
9. V. Matossian and M. Parashar. Enabling peer-to-peer interactions for scientific applications on the grid. In H. Kosch, L. Boszormenyi, and H. Hellwagner, editors, *Proceedings of the 9th International Euro-Par Conference (Euro-Par 2003)*, volume 2790 of *Lecture Notes in Computer Science*, pages 1240–1247. Springer-Verlag, August 2003.
10. R. Muralidhar and M. Parashar. A Distributed Object Infrastructure for Interaction and Steering. *Special Issue - Euro-Par 2001, Concurrency and Computation: Practice and Experience*, 15(10):957–977, 2003.
11. S. Narayanan, T. Kurc, U. Catalyurek, X. Zhang, and J. Saltz. Applying database support for large scale data driven science in distributed environments. In *Proceedings of the Fourth International Workshop on Grid Computing (Grid 2003)*, pages 141–148, Phoenix, Arizona, Nov 2003.
12. M. Parashar, G. von Laszewski, S. Verma, J. Gawor, K. Keahey, and N. Rehn. A CORBA Commodity Grid Kit. *Grid Computing Environments. Special Issue of Concurrency and Computations: Practice and Experience*, 14(13-15):1057–1074, 2002.
13. J. Saltz and et.al. Driving scientific applications by data in distributed environments. In *Dynamic Data Driven Application Systems Workshop, held jointly with ICCS 2003*, Melbourne, Australia, June 2003.
14. M. Sen and P. Stoffa. *Global Optimization Methods in Geophysical Inversion*. Advances in Exploration Geophysics 4, series editor: A.J. Berkhout. Elsevier, 1995.
15. J. C. Spall. *Introduction to stochastic search and optimization:estimation, simulation and control*. John Wiley & Sons, Inc., Publication, New Jersey, 2003.