Towards Dynamic Data-Driven Optimization of Oil Well Placement*

Manish Parashar[1], Vincent Matossian[1], Wolfgang Bangerth[2,4], Hector Klie[2], Benjamin Rutt[3], Tahsin Kurc[3], Umit Catalyurek[3], Joel Saltz[3], and Mary F. Wheeler[2]

[1] TASSL, 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, bangerth, mfw}@ices.utexas.edu
[3] Dept. of Biomedical Informatics, The Ohio State University, Ohio, USA
{rutt, umit, kurc, jsaltz}@bmi.osu.edu
[4] Institute for Geophysics, The University of Texas at Austin, Texas, USA

Abstract. The adequate location of wells in oil and environmental applications has a significant economical impact on reservoir management. However, the determination of optimal well locations is both challenging and computationally expensive. The overall goal of this research is to use the emerging Grid infrastructure to realize an autonomic dynamic data-driven self-optimizing reservoir framework. In this paper, we present the use of distributed data to dynamically drive the optimization of well placement in an oil reservoir.

1 Introduction

The locations of wells in oil and environmental applications significantly affect the productivity and environmental/economic benefits of a subsurface reservoir. However, the determination of optimal well locations is a challenging problem since it depends on geological and fluid properties as well as on economic parameters. This leads to a very large number of potential scenarios that must be evaluated using numerical reservoir simulations. Reservoir simulators are based on the numerical solution of a complex set of coupled nonlinear partial differential equations over hundreds of thousands to millions of gridblocks. The high costs of simulation make an exhaustive evaluation of all these scenarios infeasible. As a result, the well locations are traditionally determined by analyzing only a few scenarios. However, this *ad hoc* approach may often lead to incorrect decisions with a high economic impact.

* The research presented in this paper is supported in part by the National Science Foundation Grants ACI 9984357, EIA 0103674, EIA 0120934, ANI 0335244, CNS 0305495, CNS 0426354, IIS 0430826, ACI-9619020 (UC Subcontract 10152408), ANI-0330612, EIA-0121177, SBR-9873326, EIA-0121523, ACI-0203846, ACI-0130437, CCF-0342615, CNS-0406386, CNS-0426241, ACI-9982087, CNS-0305495, NPACI 10181410, DOE ASCI/ASAP via grant numbers PC295251 and 82-1052856, Lawrence Livermore National Laboratory under Grant B517095 (UC Subcontract 10184497), Ohio Board of Regents BRTTC BRTT02-0003, and DOE DE-FG03-99ER2537.

V.S. Sunderam et al. (Eds.): ICCS 2005, LNCS 3515, pp. 656 – 663, 2005.

Optimization algorithms offer the potential for a systematic exploration of a broader set of scenarios to identify optimum locations under given conditions. These algorithms, together with the experienced judgment of specialists, allow a better assessment of uncertainty and significantly reduce the risk in decision-making. Consequently, there is an increasing interest in the use of optimization algorithms for finding the optimum well location in oil industry [1,2,3,4]. However, the selection of appropriate optimization algorithms, the runtime configuration and invocation of these algorithms, and the dynamic optimization of the reservoir remains a challenging problem.

The overall goal of our research is to use the emerging Grid infrastructure [5] and its support for seamless aggregations, compositions and interactions, to enable the dynamic and autonomic data-driven optimization of oil reservoirs. In this paper we build on our autonomic reservoir management framework [6,7,8] to investigate the dynamic data-driven steering of the reservoir optimization processes for determining optimal well placement and configuration. The specific objective of this paper is to investigate how distributed data archives can be used to control and steer the optimization process to improve the quality as well as the speed of convergence.

2 Components of the Autonomic Data-Driven Oil Reservoir Framework

2.1 The Integrated Parallel Accurate Reservoir Simulator (IPARS)

IPARS represents a new approach to parallel reservoir simulator development, emphasizing modularity, code portability to many platforms, ease of integration and inter-operability 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. A key feature of IPARS is that it allows the definition of different numerical, physical, and scale models for different blocks in the domain (i.e., multi-numeric, multi-physics, and multi-scale capabilities). A more technical description of IPARS and its applications can be found in [9].

2.2 Optimization Algorithms

We use different optimization algorithms in order to capture the complexities of the application. All of these algorithms need to be able to find the optimum very efficiently, i.e. with the least number of function evaluations while not requiring gradient information as that is generally unavailable in reservoir simulators.

Very Fast Simulated Annealing (VFSA): This algorithm is a variant of simulated annealing that speeds up the process by using a variable sampling algorithm that shrinks the sampling area as the temperature parameter is decreased. This allows for a more efficient local search towards the end of the optimization process. Additionally, we use different cooling schedules for the optimization variables, see [10].

Simultaneous Perturbation Stochastic Algorithm (SPSA): The SPSA algorithm is a gradient-based algorithm; however, instead of computing the exact gradient direction, it approximates it using a random stochastic direction. Consequently, it requires only two evaluations of the objective function per iteration, regardless of the dimension of the optimization problem. This allows for a significant decrease in the cost of optimization, especially in problems with a large number of decision parameters to be estimated. The algorithm is also suitable for noisy measurements of the objective function and can be customized to perform a more global search by injecting controlled random noise (e.g., see [11]).

Gradient based: This method approximates the gradient of the objective function to derive a search direction, and moves the present iterate along this direction. In practice, this algorithm is much less efficient than other methods [12], but we implement this standard algorithm for comparison since it is a very popular and widespread method.

2.3 Querying and Subsetting of Distributed Data: STORM

An increasingly important issue in Grid computing is to enable access to and integration of data in remote repositories. An emerging approach is the *virtualization* of data sources through relational and XML models [13–15]. STORM [16] is a service-oriented middleware that supports data select and data transfer operations on scientific datasets, stored in distributed, flat files, through an object-relational database model. In STORM, data subsetting is done based on attribute values or ranges of values, and can involve user-defined filtering operations. 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.

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 can be implemented by an application developer or generated by compiler [17]. It 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 the parallel program 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.

STORM implements several optimizations to reduce the execution time of queries. These optimizations include 1) ability to execute a workflow through distributed filtering operations, and 2) execution of parallelized data transfer. Both data and task parallelism can be employed to execute 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. Data is transferred from multiple data sources to multiple destination processors in parallel by STORM data mover components.

SELECT <*Attributes*>
FROM $Dataset_1, Dataset_2, \ldots, Dataset_n$
WHERE <*Expression*> AND *Filter*(<*Attributes*>)
GROUP-BY-PROCESSOR *ComputeAttribute*(<*Attributes*>)

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

3 Autonomic Grid Middleware for Oil Reservoir Optimization

The autonomic Grid middleware supports interactions between application components, Grid services, resources (systems, CPUs, instruments, storage) and data (archives, sensors) [18]. It supports autonomic behaviors so that the interactions and feedback between simulations, services, sensors and data can be orchestrated using high-level rules, defined by expert, to navigate the parameter space and optimize the oil reservoir. Key components of the middleware are described below:

Discover [19] enables seamless access to, and peer-to-peer integration of applications, services, and resources on the Grid. The middleware substrate integrates Discover collaboratory services with the Grid services provided by the Globus Toolkit using the CORBA Commodity Grid (CORBACoG) Kit [20]. It also integrates the Pawn peer-to-peer messaging substrate [21]. Pawn enables decentralized (peer) services and applications to interact and coordinate over wide area networks. Finally, the DIOS [22] distributed object infrastructure that enables development and management of interactive 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 of applications and/or application services in an autonomic manner [23]. Detailed descriptions of the design, implementation, and evaluation of Discover components can be found in [19–23].

4 Integrated System for Data-Driven Oil Production Optimization

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

In the 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) 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., the amount of bypassed oil, the concentrations of oil and water). In a Grid environment, data analysis programs need to access data subsets on distributed storage systems [16]. This need is addressed by STORM. Figure 2 shows the performance of STORM for querying and subsetting seismic datasets. The performance numbers were obtained on a 30TB seismic dataset generated by simulations and stored on a 16-node disk-based cluster storage system, with 4.3GB/sec peak application-level bandwidth, at the Ohio Supercomputer Center. As seen from the figure, we can achieve close to 3.5GB/sec (about 75% of the peak bandwidth) bandwidth through runtime optimizations (such as distributed I/O, distributed filtering, multi-threading) implemented by STORM. The Discover autonomic Grid middleware provides the support for items 3, 4, and 5. We now discuss the use of Discover/Pawn to enable oil reservoir optimization [24].

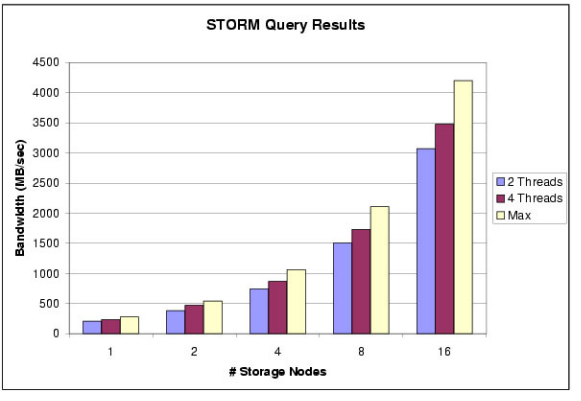

Fig. 2. Querying seismic data using STORM

The overall autonomic oil reservoir optimization 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. VFSA and SPSA); and Economic Modeling Service that uses IPARS simulation outputs and current market parameters (oil prices, costs, etc.) to compute estimated revenues for a particular reservoir configuration.

These entities 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 pervasive portals to interact with the Discover middleware 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 Optimization

algorithm is seeded using DataCutter/STORM. This seed can be obtained by querying previously executed simulations. (5) The IPARS Factory then interacts with the Discover middleware to discover and allocate resources and to configure and execute IPARS simulations. (6) 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. (7) The Optimization Service provides IPARS Factory with an improved well location, which then (8) launches new IPARS simulations with updated parameters. (9) 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.

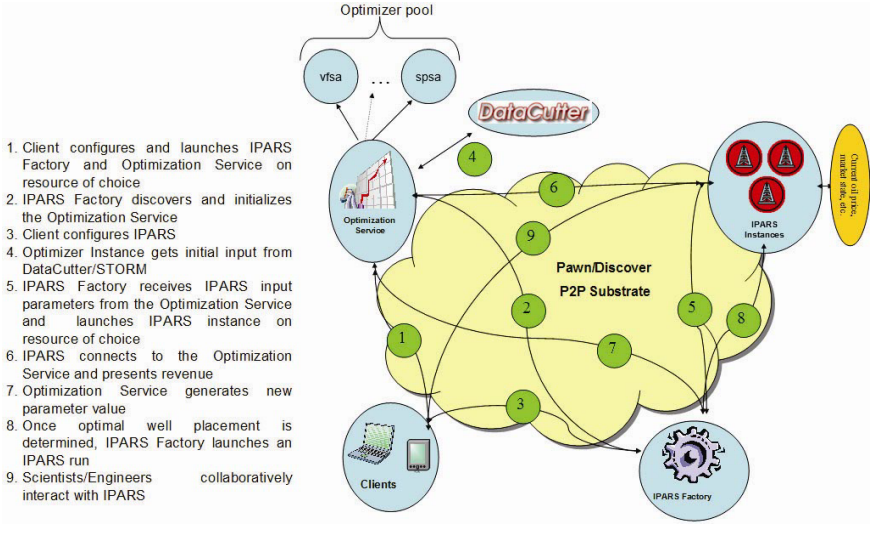

Fig. 3. Autonomic oil reservoir optimization using decentralized services

Figure 4 shows the progress of optimization of well locations using the VFSA and SPSA optimization algorithms for two different scenarios. The goal is to maximize profits for a given economic revenue objective function. The well positions plots (4(a) left and 4(b) right) show the oil field and the positions of the wells. Black circles represent fixed injection wells and a gray square at the bottom of the plot is a fixed production well. The plots also show the sequence of guesses for the position of the other production well returned by the optimization service (shown by the lines connecting the light squares), and the corresponding normalized cost value (4(a) right and 4(b) left).

The overall process described above is data-driven and autonomic in that the peers involved automatically detect sub-optimal oil production behaviors at runtime based on dynamically injected data, and orchestrate interactions among themselves to correct this behavior.

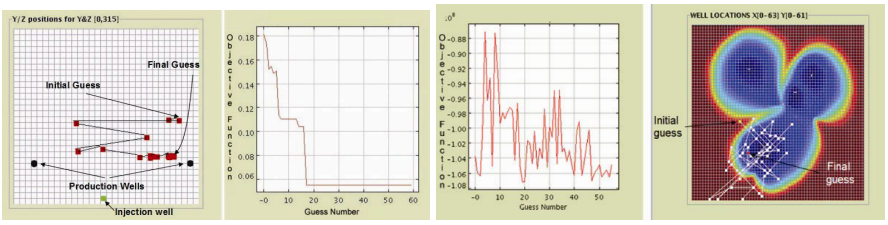

Fig. 4. Convergence history for the optimal well placement in the Grid using (a) VFSA algorithm and (b) SPSA algorithm

Further, the detection and optimization process is achieved using policies and constraints that minimize human intervention. The interactions between instances of peer services are opportunistic, based on runtime discovery and specified policies, and are not predefined.

5 Conclusion

We presented a novel infrastructure for enabling autonomic dynamic data-driven oil production management. We believe that such an infrastructure can aid in gaining better understanding of subsurface properties and decision variables, and can assist in the implementation of optimized oil production scenarios to lower infrastructure costs and maximize productivity.

References

1. Bittencourt, A.C., Horne, R.N.: Reservoir development and design optimization. In: SPE Annual Technical Conference and Exhibition, San Antonio, Texas (1997) SPE 38895.
2. Guyaguler, B., Horne, R.N.: Uncertainty assessment of well placement optimization. In: SPE Annual Technical Conference and Exhibition, New Orleans, Louisiana (2001) SPE 71625.
3. Pan, Y., Horne, R.: Improved methods for multivariate optimization of field development scheduling and well placement design. In: SPE Annual Technical Conference and Exhibition, New Orleans, Louisiana (1998) SPE 49055.
4. Yeten, B., Durlofsky, L.J., Aziz, K.: Optimization of nonconventional well type, location, and trajectory. SPE Journal **8** (2003) 200-210 SPE 86880.
5. Foster, I., Kesselman, C.: The Grid 2: Blueprint for a New Computing Infrastructure. Morgan Kaufman (2004).
6. Parashar, M., Klie, H., Catalyurek, U., Kurc, T., Matossian, V., Saltz, J., Wheeler, M.: Application of grid-enabled technologies for solving optimization problems in data-driven reservoir studies. The International Journal of Grid Computing: Theory, Methods and Applications (FGCS) **21** (2005) 19-26.
7. Matossian, V., Bhat, V., Parashar, M., Peszynska, M., Sen, M., Stoffa, P., Wheeler, M.F.: Autonomic oil reservoir optimization on the grid. Concurrency and Computation: Practice and Experience **17** (2005) 1-26.

8. Bangerth, W., Klie, H., Matossian, V., Parashar, M., Wheeler, M.F.: An autonomic reservoir framework for the stochastic optimization of well placement. Cluster Computing: The Journal of Networks, Software Tools, and Applications (2004) to appear.
9. IPARS: Integrated Parallel Reservoir Simulator, The University of Texas at Austin, http://www.ices.utexas.edu/CSM.
10. Sen, M., Stoffa, P.: Global Optimization Methods in Geophysical Inversion. Advances in Exploration Geophysics 4, editor: A.J. Berkhout. Elsevier (1995).
11. Spall, J.C.: Introduction to stochastic search and optimization, estimation, simulation and control. John Wiley & Sons, Inc., Publication, New Jersey (2003).
12. Bangerth, W., Klie, H., Wheeler, M.F., Stoffa, P.L., Sen, M.K.: On optimization algorithms for the reservoir oil well placement problem. Comp. Geosc., submitted (2004).
13. Open Grid Services Architecture Data Access and Integration. (*http://www.ogsadai.org.uk*)
14. Hastings, S., Langella, S., Oster, S., Saltz, J.: Distributed data management and integration: The mobius project. In: GGF Semantic Grid Workshop 2004. (2004) 20-38.
15. Li, X., Agrawal, G.: Using xquery for flat-file based scientific datasets. In: The 9th International Workshop on Data Base Programming Languages (DBPL). (2003).
16. Narayanan, S., Kurc, T., Catalyurek, U., Zhang, X., Saltz, J.: Applying database support for large scale data driven science in distributed environments. In: Proceedings of the Fourth International Workshop on Grid Computing (Grid 2003), Phoenix, Arizona (2003) 141-148.
17. Weng, L., Agrawal, G., Catalyurek, U., Kurc, T., Narayanan, S., Saltz, J.: An approach for automatic data virtualization. In: The Thirteenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-13). (2004).
18. Agarwal, M., Bhat, V., Li, Z., Liu, H., Matossian, V., Putty, V., Schmidt, C., Zhang, G., Parashar, M., Khargharia, B., Hariri, S.: Automate: Enabling autonomic applications on the grid. In: Autonomic Computing Workshop, The Fifth Annual International Workshop on Active Middleware Services (AMS 2003), Seattle, WA USA (2003) 365-375.
19. Mann, V., Parashar, M.: Discover: A computational collaboratory for interactive grid applications. In Berman, F., Fox, G., Hey, T., eds.: Grid Computing: Making the Global Infrastructure a Reality. John Wiley & Sons (2003) 727-744.
20. Parashar, M., von Laszewski, G., Verma, S., Gawor, J., Keahey, K., Rehn, N.: A CORBA Commodity Grid Kit. Concurrency and Computations: Practice and Experience **14** (2002) 1057-1074.
21. Matossian, V., Parashar, M.: Enabling peer-to-peer interactions for scientific applications on the grid. In Kosch, H., Boszormenyi, L., Hellwagner, H., eds.: Proceedings of the 9th International Euro-Par Conference. Volume 2790 of Lecture Notes in Computer Science, Springer-Verlag (2003) 1240-1247.
22. Muralidhar, R., Parashar, M.: A Distributed Object Infrastructure for Interaction and Steering. Special Issue - Euro-Par 2001, Concurrency and Computation: Practice and Experience **15** (2003) 957-977.
23. Liu, H., Parashar, M.: Dios++: A framework for rule-based autonomic management of distributed scientific applications. In Kosch, H., Boszormenyi, L., Hellwagner, H., eds.: Proceedings of the 9th International Euro-Par Conference. Volume 2790 of Lecture Notes in Computer Science, Springer-Verlag (2003) 66-73.
24. Matossian, V., Parashar, M.: Autonomic optimization of an oil reservoir using decentralized services. In: Proceedings of the 1st International Workshop on The Challenges for Large Applications in Distributed Environments (CLADE 2003), Computer Society Press (2003) 2-9.