Available online at www.sciencedirect.com

SCIENCE DIRECT®

Future Generation Computer Systems 21 (2005) 19–26

FGCS
FUTURE GENERATION COMPUTER SYSTEMS

www.elsevier.com/locate/future

Application of Grid-enabled technologies for solving optimization problems in data-driven reservoir studies

Manish Parashar[a,*], Hector Klie[b], Umit Catalyurek[d], Tahsin Kurc[d], Wolfgang Bangerth[b,c], Vincent Matossian[a], Joel Saltz[d], Mary F. Wheeler[b]

[a] *TASSL, Department of Electrical and Computer Engineering, Rutgers, The State University of New Jersey, NJ, USA*
[b] *CSM, ICES, The University of Texas at Austin, TX, USA*
[c] *Institute for Geophysics, The University of Texas at Austin, TX, USA*
[d] *Department of Biomedical Informatics, The Ohio State University, OH, USA*

Available online 2 November 2004

Abstract

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

Keywords: Grid computing; Optimization algorithms; Distributed database

1. Introduction

The ultimate goal of reservoir modeling is to generate both good estimates of reservoir parameters and reliable predictions of oil production to optimize return on investment from a given reservoir. This process is performed by the use of numerical simulators that represent the multiphase fluid flow phenomenon under the subsurface. However, little use has been made of reservoir simulations coupled with systematic optimization techniques. The main advantage of applying these mathematical tools to the 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, which operate by searching a large space of oil production and reservoir parameters. The *dynamic, data-driven application systems* (DDDAS) paradigm provides a viable mechanism to address this issue. A main feature of DDDAS is the on-the-fly interaction between and integration of numerical models and data from simulations or field measurements. The integration of data and numerical models through DDDAS allows for a more efficient search of the parameter space.

* Corresponding author.

E-mail addresses: parashar@caip.rutgers.edu (M. Parashar), kile@ices.utexas.edu (H. Klie), umit@bmi.osu.edu (U. Catalyurek), kurc@bmi.osu.edu (T. Kurc), bangerth@ices.utexas.edu (W. Bangerth), vincentm@caip.rutgers.edu (V. Matossian), jsaltz@bmi.osu.edu (J.Saltz), mfw@ices.utexas.edu (M.F. Wheeler).

0167-739X/$ – see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2004.09.028

Several obstacles, however, need to be addressed for a successful application of DDDAS. The first one is the computational time required to complete simulations of complex, large scale reservoir models. Another challenge is to implement the capability to manage and navigate multi-terabyte datasets from simulations and field measurements. Optimization strategies normally evaluate hundreds or even thousands of scenarios (each requiring 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 [1,2].

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 to access hardware, software, and information resources [3,2].

In our previous work, we described a suite of tools and middleware that enable execution and analysis of large, distributed collections of simulations and datasets [4,5]. In this paper, we present the infrastructure for solving optimization problems in dynamic, data-driven reservoir simulations in the Grid. The infrastructure builds on three 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/Pawn) for service composition, execution, and collaboration. We describe these components and their application in autonomic data-driven management of the oil production process [3,6].

2. Computational Grid components

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 interoperability 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, multiphysics, and multi-scale capabilities). A more technical description of IPARS and its applications can be found in [7].

2.2. Optimization algorithms

2.2.1. Very fast simulated annealing (VFSA)

This algorithm is a simulated annealing variant that speeds up the process by allowing a larger sampling at the beginning and a much narrower sampling at later stages. This is achieved by using a Cauchy like distribution. The second appealing feature of VFSA 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., see [8]).

2.2.2. Simultaneous perturbation stochastic algorithm (SPSA)

The novelty of the SPSA algorithm is the underlying derivative approximation that requires only one or two evaluations of the objective function regardless of the dimension of the optimization problem. In other words, it does not require an accurate gradient computation. This feature allows for a significant decrease in the cost of optimization, especially in problems with a large number of decision parameters to be estimated. This algorithm is suitable for noisy measurements of the objective function and can be customized to perform a more global search by the injection of controlled random noise (e.g., see [9]).

2.2.3. Gradient based

These methods essentially use the approximated gradient of the objective function 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 [10] and finite-difference stochastic approximation (FDSA) methods [9] are representative examples.

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 [11–13]. STORM (formerly called GridDB-Lite) [14] 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 Fig. 1. The $\langle Expression \rangle$ 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 [15]. 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 is implemented using DataCutter [16]. DataCutter is a component-based framework [17] that implements a filter-stream model for data processing. In this model, data is pushed from data sources to destination processors through a network of application-defined processing components. Using DataCutter runtime support, 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 by STORM data mover components. Data movers can be instantiated on multiple storage units and destination processors to achieve parallelism during data transfer.

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.

4. An autonomic Grid middleware for oil reservoir optimization

Discover [18] 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 [19]. It also integrates the Pawn peer-to-peer messaging substrate [20]. Pawn enables decentralized (peer) services and applications to interact and coordinate over wide area networks. Finally, the DIOS [21] 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 [22]. Detailed descriptions of the design, implementation, and evaluation of Discover components can be found in [18–22].

5. Putting it together: data-driven oil production optimization

The oil production optimization process involves (1) the use of an integrated multi-block 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 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) 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 [4,14]. This need is addressed 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 [2].

The overall autonomic oil reservoir optimization scenario is illustrated in Fig. 2. 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.

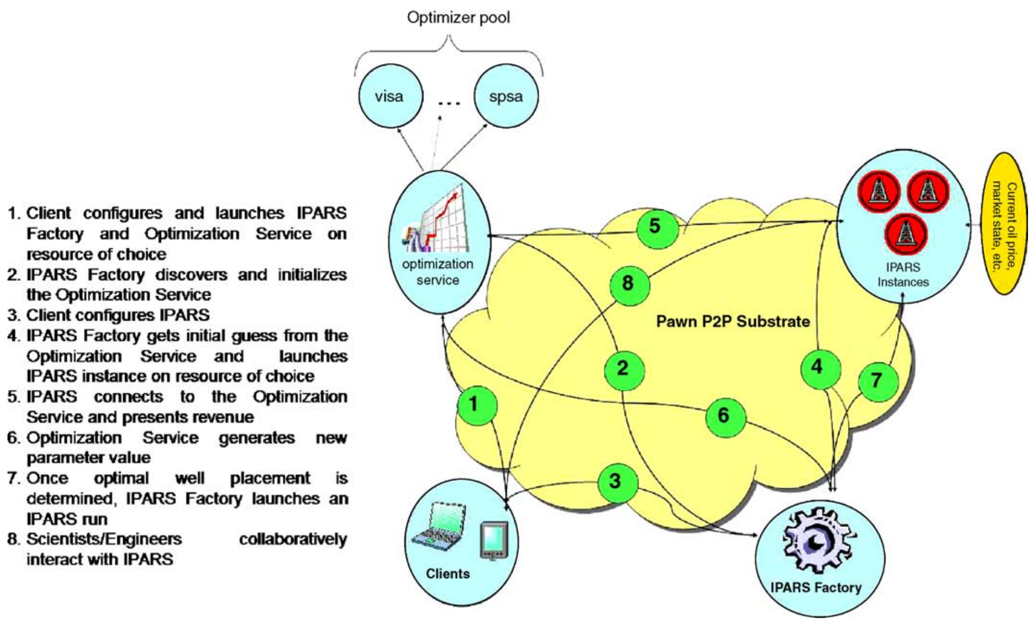


Fig. 2. Autonomic oil reservoir optimization using decentralized services.

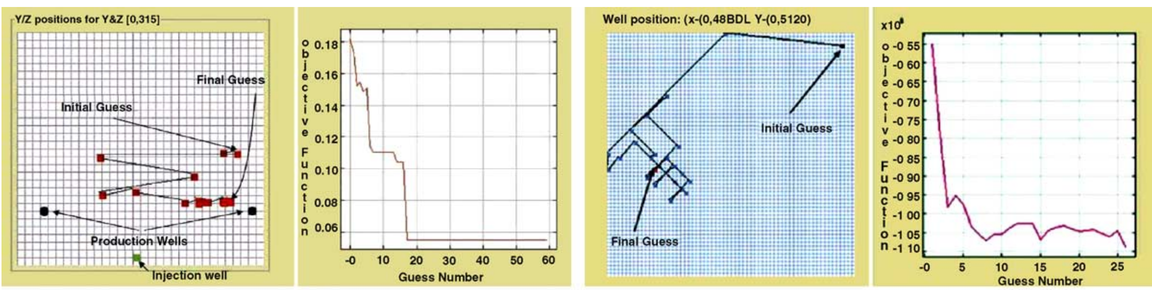
These entities dynamically discover and interact with one another as peers to achieve the overall application objectives. Fig. 2 illustrates the key interactions involved. (1) The experts use pervasive portals to interact with the Discover middleware and the Globus Grid services to discover and allocate 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 IPARS Factory then interacts with the Discover middleware 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 an improved well location, which then (7) launches new IPARS simulations with updated 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.

Fig. 3 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 (on the left in Fig. 3(a) and (b)) 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 (plots on the right in Fig. 3(a) and (b)).

6. Policy-driven optimization

A key objective of this research is to formulate policies that can be used by the autonomic self-optimizing reservoir framework to discover, select, configure, and invoke appropriate optimization services to determine optimal well locations. The choice of optimization service depends on the size and nature of the reservoir. The SPSA algorithm is suited for larger reservoirs with relatively smooth characteristics. In case of reservoirs with many randomly distributed maxima and minima, the VFSA algorithm can be employed during the initial optimization phase. Once convergence slows down, VFSA can be replaced by SPSA. Alternate optimization schemes (e.g., genetic algorithms, local methods such as Newton) can also be used if convergence breaks down. Similarly, policies can also be used to manage the behavior of the reservoir simulator. For example, the policy may monitor convergence of the optimizer and as it approaches the solution, it may use a finer mesh and/or smaller timesteps. The policy may even attempt to activate other numerical algorithms (e.g., time discretization schemes, solvers) or physical models (e.g., one-, two-, or three-phase flow, geomechanical).

In an alternative scenario, policies may be defined to enable various optimizers to execute concurrently on dynamically acquired Grid resources, and select the



(a) (b)

Fig. 3. Convergence history for the optimal well placement in the Grid using (a) VFSA algorithm and (b) SPSA algorithm.

best well location among these based on some metric (e.g., estimated revenue, time or cost of completion). This aspect is important for speeding up the search, or for studying the effects of parameters that were not included at the start of the optimization. The autonomic reservoir framework presented enables the decoupling of services and the separation of policy and mechanism. This allows external policies, such as those outlined above, to be dynamically defined and used to manage the behavior of the components/services, and to orchestrate interactions between them to achieve overall optimization goals for the reservoir.

It is worth adding that the present paradigm has the potential to exploit several levels of parallelism. That is, for different geological and economical scenarios (equally probable models) the optimization can be carried out independently with different initial guesses for the well location. Each realization and well location is evaluated by means of the parallel reservoir simulator IPARS, giving rise to a three-level hierarchy of totally independent parallel tasks.

7. Conclusion

In this paper, we presented an infrastructure and its components to support the autonomic oil production management process. Use of this infrastructure to implement 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 R. Martino and M. Peszyńska for their help in executing the VFSA-based experiments. This work is partly supported by 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, CNS-0305495, NPACI 10181410, ACI-9984357, EIA-0103674 and EIA-0120934, 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.

References

[1] V.M. Johnson, L.L. Rogers, Applying soft computing methods to improve the cómputational tractability of a subsurface simulation–optimization problem, J. Petroleum Sci. Eng. 29 (2001) 153–175.

[2] V. Matossian, M. Parashar, 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, pp. 2–9.

[3] V. Bhat, V. Matossian, M. Parashar, M. Peszynska, M. Sen, P. Stoffa, M.F. Wheeler, Autonomic oil reservoir optimization on the grid, Concurrency Comput.: Pract. Exp., 2004, to appear.

[4] J. Saltz, et al., Driving scientific applications by data in distributed environments, in: Dynamic Data Driven Application Systems Workshop, held jointly with ICCS 2003, Melbourne, Australia, 2003.

[5] M. Parashar, R. Muralidhar, W. Lee, M.F. Wheeler, D. Arnold, J. Dongarra, Enabling interactive oil reservoir simulations on the grid, Concurrency Comput.: Pract. Exp., 2004, to appear.

[6] W. Bangerth, H. Klie, V. Matossian, M. Parashar, M.F. Wheeler, An autonomic reservoir framework for the stochastic optimization of well placement, Cluster Comput.: J. Networks Software Tools Appl., 2004, to appear.

[7] IPARS: Integrated Parallel Reservoir Simulator, The University of Texas at Austin, http://www.ices.utexas.edu/CSM.

[8] M. Sen, P. Stoffa, Global optimization methods in geophysical inversion, in: A.J. Berkhout (Ed.), Advances in Exploration Geophysics, Elsevier, vol. 4, 1995.

[9] J.C. Spall, Introduction to stochastic search and optimization:estimation, simulation and control, John Wiley & Sons Inc., Publication, New Jersey, 2003.

[10] J. Dennnis, R. Schnabel, Numerical Methods for Unconstrained Minimization and nonlinear equations, Prentice-Hall, Englewood Cliffs, New York, 1983, reprinted by SIAM, Classics Appl. Math., 1996.

[11] Open Grid Services Architecture Data Access and Integration, http://www.ogsadai.org.uk.

[12] S. Hastings, S. Langella, S. Oster, J. Saltz, Distributed data management and integration: the mobius project, GGF Semantic Grid Workshop 2004, 2004, pp. 20–38.

[13] X. Li, G. Agrawal, Using xquery for flat-file based scientific datasets, in: Proceedings of the 9th International Workshop on Data Base Programming Languages (DBPL), 2003.

[14] S. Narayanan, T. Kurc, U. Catalyurek, X. Zhang, 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), Phoenix, Arizona, 2003, pp. 141–148.

[15] L. Weng, G. Agrawal, U. Catalyurek, T. Kurc, S. Narayanan, J. Saltz, An approach for automatic data virtualization, in: Proceedings of the Thirteenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-13), 2004.

[16] M.D. Beynon, T. Kurc, U. Catalyurek, C. Chang, A. Sussman, J. Saltz, Distributed processing of very large datasets with DataCutter, Parallel Comput. 27 (11) (2001) 1457–1478.

[17] D.E. Bernholdt, W.R. Elwasif, J.A. Kohl, T.G. Epperly, A component architecture for high-performance computing, in: Proceedings of the Workshop on Performance Optimization via High-Level Languages and Libraries (POHLL-02), ACM Press, 2002.

[18] V. Mann, M. Parashar, Discover: a computational collaboratory for interactive grid applications, in: F. Berman, G. Fox, T. Hey (Eds.), Grid Computing: Making the Global Infrastructure a Reality, John Wiley & Sons, 2003, pp. 727–744.

[19] M. Parashar, G. von Laszewski, S. Verma, J. Gawor, K. Keahey, N. Rehn, A CORBA commodity Grid kit, Concurrency and Computations: Practice and Experience 14 (13–15) (2002) 1057–1074.

[20] V. Matossian, M. Parashar, Enabling peer-to-peer interactions for scientific applications on the grid, in: H. Kosch, L. Boszormenyi, H. Hellwagner (Eds.), Proceedings of the 9th International Euro-Par Conference, vol. 2790 of Lecture Notes in Computer Science, Springer-Verlag, 2003, pp. 1240–1247.

[21] R. Muralidhar, M. Parashar, A distributed object infrastructure for interaction and steering, Special Issue–Euro-Par 2001, Concurrency Comput.: Pract. Exp. 15 (10) (2003) 957–977.

[22] H. Liu, M. Parashar, Dios++: a framework for rule-based autonomic management of distributed scientific applications, in: H. Kosch, L. Boszormenyi, H. Hellwagner (Eds.), Proceedings of the 9th International Euro-Par Conference, vol. 2790 of Lecture Notes in Computer Science, Springer-Verlag, 2003, pp. 66–73.

Manish Parashar is associate professor of electrical and computer engineering at Rutgers University, where he also is director of the Applied Software Systems Laboratory. He received a BE degree in electronics and telecommunications from Bombay University, India in 1988, and MS and PhD degrees in computer engineering from Syracuse University in 1994. He has received the NSF CAREER Award (1999) and the Enrico Fermi Scholarship from Argonne National Laboratory (1996). His current research interests include autonomic computing, parallel, distributed and Grid computing, networking, scientific computing, and software engineering. Manish is a member of the executive committee of the IEEE Computer Society Technical Committee on Parallel Processing (TCPP), part of the IEEE Computer Society Distinguished Visitor Program (2004–2006), and a member of ACM. He is also the co-founder of the IEEE International Conference on Autonomic Computing (ICAC). Manish has co-authored over 130 technical papers in international journals and conferences, has co-authored/edited 5 books/proceedings, and has contributed to several others in the area of parallel and distributed computing.

Hector Klie got a PhD degree in computational science and engineering at Rice University, 1996, he completed his master and undergraduated degrees in computer science at the Simon Bolivar University, Venezuela in 1991 and 1989, respectively. Hector Klie's main research interests are in the mathematics, numerical analysis, and high performance computational issues related to the solution of transport and flow of porous media equations. He devotes particular interest to the development of parallel iterative methods for the solution of large scale linear, nonlinear and coupled problems. He is currently pursuing novel optimization approaches for oil reservoir simulation coupled with seismic, geomechanics and other different physical models (multiphysics) for instrumented and automatic oil field management. He also has extensive industrial experience in various geophysical problems such as tomography, wave modeling in anisotropic media and converted wave data processing. He has written several papers in these geoscientific research areas. He currently holds the position of associate director and senior research associate in the Center for Subsurface Modeling at the Institute of Computational Science and Engineering at The University of Texas at Austin.

Umit Catalyurek is an assistant professor in the Department of Biomedical Informatics at The Ohio State University. His research interests include graph and hypergraph partitioning algorithms, grid computing, and runtime systems and algorithms for high-performance and data-intensive computing. He received his PhD, MS and BS in computer engineering and information science from Bilkent University, Turkey, in 2000, 1994 and 1992, respectively.

Tahsin Kurc is an assistant professor in the Department of Biomedical Informatics at the Ohio State University. His research interests include runtime systems for data-intensive computing in parallel and distributed environments, and scientific visualization on parallel computers. He received his PhD in computer science from Bilkent University, Turkey, in 1997 and his BS in electrical and electronics engineering from Middle East Technical University, Turkey, in 1989.

Wolfgang Bangerth is a postdoctoral research fellow at both the Institute for Computational Engineering and Sciences, and the Institute for Geophyics, at The University of Texas at Austin. He obtained his PhD in applied mathematics from the University of Heidelberg, Germany in 2002. He is the project leader for the deal.II finite element library (http://www.dealii.org). Wolgang is a member of the ACM.

Vincent Matossian obtained a masters in applied physics from the French Université Pierre et Marie Curie. Vincent is currently pursuing a PhD degree in distributed systems at the Department of Electrical and Computer Engineering at Rutgers University under the guidance of Manish Parashar. His research interests include information discovery and ad-hoc communication paradigms in decentralized systems.

Joel Saltz is professor and chair of the Department of Biomedical Informatics, professor in the Department of Computer and Information Systems and a senior fellow of the Ohio Supercomputer Center. Prior to coming to Ohio State, Dr. Saltz was professor of Pathology and Informatics in the Department of Pathology at Johns Hopkins Medical School and professor in the Department of Computer Science at the University of Maryland. He received his MD and PhD in computer science from Duke University in 1985 and 1986, respectively. He earned his BS in mathematics and physics from University of Michigan in 1978. His research interests are in the development of systems software, databases and compilers for the management, processing and exploration of very large datasets.

Dr. Mary Fanett Wheeler has directed the Center since its inception as the Subsurface Modeling Group at Rice University from 1990 to 1995, when the group moved to The University of Texas at Austin. She has directed over 25 PhD graduates who are presently employed in the petroleum and environmental industries, universities, and government laboratories. Her primary research interest is in the numerical solutions of partial differential systems with applications to flow in porous media, geomechanics, surface flow, and parallel computation. Her numerical work includes formulation, analysis and implementation of finite-difference/finite-element discretization schemes for nonlinear, coupled PDE's as well as domain decomposition iterative solution methods. Dr. Wheeler has published more than 100 technical papers and edited seven books. She is currently an editor of six technical journals and managing editor of Computational Geosciences. In 1998 she was elected to the National Academy of Engineering and currently holds the Ernest and Virginia Cockrell Chair in Engineering. She is currently professor in the Aerospace Engineering and Engineering Mechanics and in Petroleum and Geosystems Engineering.