Simulating and Visualizing the Human Arterial System on the TeraGrid

Suchuan Dong,[1] Joseph Insley,[4] Nicholas T. Karonis,[2,4]
Michael E. Papka,[3,4] Justin Binns,[4] and George Karniadakis[1]

[1] Division of Applied Mathematics, Brown University, Providence, RI 02863
[2] Department of Computer Science, Northern Illinois University, DeKalb, IL 60115
[3]Department of Computer Science, University of Chicago, Chicago, IL 60137
[4]Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439

Abstract

We present a Grid solution to a grand challenge problem, the simulation and visualization of the human arterial system. We implemented our simulation and visualization system on the National Science Foundation's TeraGrid and demonstrated it at the *i*Grid 2005 conference in San Diego, California. We discuss our experience in running on a computational Grid and present observations and suggestions for improving similar experiences.

1. Simulating the Human Arterial System

The present study is motivated by a grand challenge problem in biomechanics— the simulation of blood flow in the *entire* human arterial tree—whose solution is not feasible with conventional supercomputers. This problem presents significant fundamental and application challenges, but a solution will have profound scientific and societal impacts.

The human arterial tree simulation problem originates from the widely accepted causal relationship between blood flow and the formation of arterial disease such as atherosclerotic plaques. These disease conditions preferentially develop in separated and recirculating flow regions such as arterial branches and bifurcations. Interactions of blood flow in the human arterial system can occur between different scales, or at similar scales in different regions, of the vascular system. At the largest scale, the human arterial system is coupled through the wavelike nature of the pulse information traveling from the heart into elastic arteries. Surgical interventions, such as bypass grafts, alter the wave reflections, which in turn can modify the flow waveforms at seemingly remote locations. Subsequently, the modification of a local waveform can lead to the onset of undesirable wall stresses, thereby possibly starting another pathological event.

Modeling of the three-dimensional unsteady fluid dynamics within sites of interest such as arterial branches requires enormous processing power. What makes this type of application amenable to Grid computing is that the waveform coupling between the sites of interest can be reasonably modeled by a reduced set of one-dimensional equations that capture the cross-sectional area and sectional velocity properties [7]. One can therefore simulate the entire arterial tree using a hybrid approach based on a reduced set of one-dimensional equations for the overall system and detailed 3D Navier-Stokes equations at arterial branches and bifurcations. To capture the flow dynamics in an artery bifurcation reasonably well, the grid resolution typically requires a mesh of 70,000 to 200,000 finite elements of high order; here we use spectral elements with a spectral polynomial order of 10 to 12 on each element [3]. The human arterial tree model in Figure 1 contains the largest 55 arteries in the human body with 27 artery bifurcations. The inclusion of all 27 artery bifurcations in the simulation with the above grid resolutions requires a total memory of 3 to 7 terabytes, which is beyond the current capacity of any single supercomputing site available to the open research community in the United States. At present, the National Science Foundation's TeraGrid has a combined processing power of nearly 50 teraflops and 1.5 petabytes of online storage and is connected over a 30 Gb/s intersite network [1]. Equipped with MPICH-G2 [4] and the Globus-family of Grid services, the TeraGrid makes simulations at these resolutions possible.

2. Simulating and Visualizing the Human Arterial System on the TeraGrid

Computational Grids offer unprecedented computational power and storage capacity and thus have opened the possibility of solving problems that were previously not possible on even the largest single computational resources [2]. These opportunities notwithstanding, developing Grid applications that run efficiently remains a challenge because of the heterogeneity of networks and system architectures inherent in Grid environments. We describe here our solution to the arterial simulation problem and our visualization pipeline as they were both executed on the TeraGrid. We start with a brief description of the middleware we used.

2.1 Spectral Element Code NEKTAR and MPICH-G2

Our solution uses a high-order CFD code, NekTar, based on a spectral/hp element method to discretize in space and a semi-implicit scheme in time. The mesh consists of structured or unstructured grids or a combination of both, similar to those used in standard finite element and finite volume methods. Flow variables are represented in terms of Jacobi polynomial expansions. This provides multiresolution, a way of hierarchically refining the numerical solution by increasing the order of the expansion (p-refinement) within each element without the need to regenerate the mesh, thus avoiding significant overhead.

We also use MPICH-G2, an MPI library that extends the MPICH implementation of MPI to use services provided by the Globus Toolkit. The library exploits MPI constructs for application-level adaptation to the network topology. MPICH-G2 selects the most efficient communication method possible between two processes, using vendor-supplied MPI if available, or TCP otherwise. It also uses information in the Globus Resource Specification Language script to create multilevel clustering of the processes based on the underlying network topology and makes that information available to applications through attributes in MPI communicators.

2.2 Human Arterial Tree Simulation on the TeraGrid

In the hybrid approach for human arterial tree simulations we use a system of reduced conservation equations to capture the wavelike interaction of the blood flow. Within a human arterial tree the solution to these equations captures the wave propagation as flow is ejected from the heart and the subsequent reflection of waves at arterial branches and the peripheral vasculature. Disease onset occurs at arterial branches, or bifurcations, and so we use 3D unsteady Navier-Stokes equations to produce detailed information at those sites. Therefore, the blood flow simulation of the human arterial network consists of an overall 1D simulation through the full arterial tree and detailed 3D simulations of arterial bifurcations. The overall simulation of the arterial tree is loosely coupled through 1D variables only.

1D simulation of the full arterial tree is conducted on a single TeraGrid site (master site). This is feasible owing to the limited computational resources required by the 1D model. 3D simulations of arterial bifurcations are conducted at different TeraGrid sites, including the master site itself (see Figure 1). At each time step the 1D results, blood flow velocity and pressure, as depicted in Figures 2(a) through 2(d), feed the full 3D simulations with inflow conditions. Therefore, cross-site communication involves only broadcasts of 1D data from the master site (i.e., the 1D group described below) to the other TeraGrid sites. The full 3D simulation of each bifurcation is conducted within a TeraGrid site by using NekTar and MPICH-G2 through domain decomposition.

The simulation's processes are arranged into three types of process groups (i.e., MPI communicators): (1) a single group responsible for 1D model computations (1D group), (2) groups responsible for 3D simulations with each corresponding to a artery bifurcation (bifurcation group), and (3) groups responsible for cross-site boundary condition communications from 1D model to 3D simulations with again one per bifurcation. The 1D and bifurcation groups partition the processes. We used the network topology information made available by MPICH-G2 to dynamically create process groups to ensure that all processes from the 1D group and each bifurcation group execute on the same TeraGrid site. Each BC group, one per bifurcation, includes all the processes from the corresponding bifurcation group and a process from the 1D group that holds the 1D result for that particular bifurcation. Cross-site communications involve only processes in the BC groups as a broadcast from the 1D process to the bifurcation processes, which, using MPICH-G2's network topology-aware collective operations, results in only a single cross-site message.

The one-way coupling (3D simulations depend on the 1D data, but not vice versa) and the computation time difference between 1D and 3D simulations (1D simulation is significantly faster than 3D simulations) are exploited to overlap cross-site communications with in-site computations. Specifically, in a cross-site communication the master site sends 1D data that spans M time steps to 3D simulations using MPI nonblocking calls, where M is a tunable parameter depending on the relative speed of 1D and 3D simulations. Therefore, for 3D simulations the in-site computation of the current M time steps overlaps with the cross-site receive of the next M steps of inflow data. For the 1D simulation the in-site computation of the current M time steps overlaps with the cross-site send of the previous M steps of 1D data. The 3D simulations lag behind the 1D simulation by about M time steps.

2.3 The Visualization Pipeline

The visualization pipeline was designed with several goals in mind and builds from previous visualization work [5,6]. First, it needed to provide high-level feedback about the state of the on-going simulation in as close to real time as possible. Second, it had to enable users to investigate particular areas of interest in greater detail. Third, and perhaps most important, the system had to have little impact on the performance and stability of the running simulation; that is, any failure in the visualization should not cause the simulation to fail as well. Since the design of the visualization system was largely driven by the characteristics of data that feeds it (i.e., data heterogeneity and rate), we start with a brief description of that data.

First is the data to visualize the 1D simulation, which is a single value for each of the 55 arteries. Because this is a small amount of data, it can be transferred and visualized at every time step. We also consider two types of 3D simulation data. The high-resolution 3D data consists of values at each point in the 3D mesh. Since the amount of this data can be quite large, it is not practical to transfer this data every time step, as this process would negatively impact the simulation. Instead, the data is transferred and visualized once every 5-time steps. The low-resolution 3D data consists only of the data on the boundary of the bifurcations. This data is considerably smaller and can therefore be transferred and visualized more frequently; we do so every 3 time steps.

The visualization pipeline is separated into three independent components. The first component is the visualization server (VS), which runs as a single MPI application and is the input to the visualization pipeline. It establishes MPI connections to the simulation from which it receives data and makes that data available to the high- and low-resolution clients that are both downstream in the visualization pipeline. The VS includes one process for the 1D simulation and one process for each of the 3D bifurcation simulations. Using MPICH-G2's accept/connect functionality, each VS process opens a port (i.e., calls MPI_Open_port) and waits for a simulation process to connect (i.e., call MPI_Comm_connect). After the connection is established, the 1D VS process writes the data it receives from the 1D simulation process to disk and updates a separate local file to report the time step and the simulation time of the 1D data that was just written to disk. Similarly, the 3D VS processes do the same for both the high- and low-resolution data they receive from the 3D simulation processes.

The other two components are the high- and low-resolution visualization clients (VCs), which sit side by side in the visualization pipeline (i.e., one is not downstream of the other) and are both consumers of the VS output. The low-resolution VC is a single MPI application with one process for the 1D data produced by the VS and one process for each of the 3D bifurcation simulations for the low-resolution data (i.e., boundary data) also produced by the VS. Each 3D bifurcation simulation has a high-resolution VC; each executing as its own MPI application, and each consuming the high-resolution data produced by the 3D VS processes. Both the low- and the high-resolution VCs take the data produced by the VS and create images. The VS and VCs all share the same file system and use files, rather than MPI, to communicate. Decoupling the VCs from the simulation in this way provides a measure of fault tolerance to the simulation.

3. Running Demonstrations at *i*Grid 2005

In September 2005 we ran the simulation and visualization pipeline described in the previous section as demonstrations at the *i*Grid 2005 conference in San Diego, California. We used the TeraGrid for both the simulation and the image generation and then sent the images to the conference showroom floor for visualization. We describe here how the simulation and the visualization pipeline were distributed across the TeraGrid's facilities. We also present performance results that demonstrate the efficacy of our simulation solution.

3.1 Details of Our TeraGrid Demonstration

At the core of our demonstration was the human arterial simulation as it executed on TeraGrid sites across the United States. Driven by the 1D computation, the 3D simulations included six artery bifurcations: carotid, aorta-superior mesenteric, femoral-genicular (left and right legs), femoral-profunda, and profunda-perforating branch. The 3D simulations of the first four bifurcations and the 1D model were computed at National Center for Supercomputing Applications (NCSA) and the last two 3D simulations at the San Diego Supercomputing Center (SDSC). We used a total of 200 processes spread across NCSA and SDSC, with two processes for the 1D model and 33 processes for each artery bifurcation. Among the 33 processes for an artery bifurcation, one was a process dedicated to transferring the flow data of that bifurcation in real time to the corresponding visualization-server process running at the University of Chicago/Argonne National Laboratory's (UC/ANL) TeraGrid facilities (described below), and the other 32 processes performed 3D computations. Of the two processes for the 1D model, one sent data to the VS, and the other performed the 1D computations. The pressure data on the artery walls (i.e., the low resolution data) was sent to VS every three time steps, while the volume flow data (i.e., three velocity components and the pressure) was sent every five time steps.

The visualization pipeline ran at UC/ANL's TeraGrid facilities and we had a viewer client running on the conference showroom floor. The VS ran on the TeraGrid IA64 compute nodes at UC/ANL. The high- and low-resolution visualization clients ran on the TeraGrid IA32 visualization nodes at UC/ANL and on a machine on the conference's showroom floor. Figure 3, which depicts the normalized pressure of the blood flow in an arterial bifurcation from the right leg, is one of the 3D images we generated from the data resulting from one of our conference runs.

3.2 Performance Results

We examine here the performance of our artery simulation by comparing multisite computations to single-site computations in controlled experiments conducted on the TeraGrid before the conference and without connecting the simulation to the visualization pipeline. Figures 4(a) and 4(b) depict the scalability of a fixed-size problem run up to nearly 400 processes and nearly 200 processes, respectively. Two of the lines in Figure 4(a) represent single-site runs at NCSA and the Pittsburgh Supercomputing Center (PSC), and the remaining three lines represent the same fixed-size problem spread evenly across the three TeraGrid sites NCSA, SDSC, and PSC as reported by processes running at each of the three sites, respectively. Likewise, Figure 4(b) depicts one line representing a single site run at NCSA and three lines solving the same fixed-size problem spread evenly across NCSA, SDSC, and UC/ANL. Since the UC/ANL TeraGrid cluster has a limited number of processors, we had to limit the total number of processes in the experiments shown in Figure 4(b) in order to keep the distribution of processes even across the three TeraGrid sites. As these figures show, no appreciable difference in scalability occurs as the application moves from a single site to up to three sites across the Grid. These results demonstrate that our solution to the human arterial simulation problem effectively tolerates the heterogeneous network characteristics found in computational Grids, specifically the significant differences in intercluster vs. intracluster latency and bandwidth that often means the demise of Grid applications.

We also studied the scalability of our application as it used the TeraGrid to solve problems of increasing size. Figure 5 depicts two execution cycles of our human arterial application, one that was always started from SDSC and a second that was always started from NCSA. Each line starts with a problem solved at a single site. As we add more sites, we add the same number of processors at each, thus increasing the problem size by a factor equal to the original problem size (i.e., we double the problem size when running at two sites and triple the problem on three sites). We observe essentially linear scalability as we increase the problem size proportionally with the number of sites and compute power, once again demonstrating our application's ability to effectively scale in a heterogeneous Grid environment.

4. Case Study in Cross-Site Grid Computing

In the weeks leading up to the *i*Grid 2005 conference we worked with system administrators at PSC and the Texas Advanced Computing Center (TACC) TeraGrid sites to successfully deploy the Globus Toolkit and MPICH-G2. Despite our many attempts during the week of the conference to simulate eight bifurcations across SDSC, NCSA, PSC, and TACC, ultimately we simulated only six bifurcations across NCSA and SDSC. We describe here the problems we experienced.

One set of problems we encountered was unveiled by the fact that until arriving in San Diego the day before the conference, we did not have access to the new computers that were sitting on the conference showroom floor. Thus, we had to install the Globus Toolkit and MPICH-G2 in one day. Situations like these are often unavoidable for conference demonstrations and typically result in last-minute debugging efforts (as they did for us). This process becomes particularly difficult when attempting to install and deploy software as complicated as the Globus Toolkit. In such situations, one relies entirely on the middleware's error messages and the advice of middleware gurus. Unfortunately, the error messages we had to work with were at best not helpful and at worst misleading. For example, what turned out to be a clock problem was repeatedly reported by Globus as a security problem (with no mention of time or expiration). After we had spent a lot of time and effort checking certificates, proxies, and the like and had had many phone conversations with Globus gurus, it was ultimately determined that the clock was not properly set on the new machine nor was the new machine's name registered in the Domain Name Service. (We note that the problem of error messages that are misleading or simply not helpful is not limited to new installations. The same problem also surfaces when things go wrong for cross-site grid applications that are running on systems in which the Grid middleware has been installed and running for a long time.)

Another set of problems stemmed from the TeraGrid's inability to meet all the needs of cross-site Grid applications. First, there is the hidden IP problem. Much of the time working with the PSC and TACC system administrators in the weeks prior to the *i*Grid 2005 conference was spent in solving the problems introduced by the network configuration of their computing clusters. Both PSC and TACC have large computing facilities and, like many other sites with clusters of similar size, opted to configure their compute nodes with hidden IP addresses. While this configuration decision is rarely problematic for single-site applications, it is obviously lethal for cross-site applications. Despite the fact that prior to the conference we had solved problems in the Globus Toolkit and MPICH-G2 brought forth by hidden IP addresses and had successfully run cross-site MPICH-G2 applications across PSC and TACC, other problems associated with the hidden IP addresses were never resolved (e.g., wget never worked) and therefore prevented us from running at these sites. Second, there is the problem of co-scheduling. Cross-site Grid applications (e.g., MPICH-G2 applications) require that TeraGrid facilities from multiple locations be co-scheduled. Currently this process is done by contacting system administrators at each site to arrange a co-scheduled reservation. Not only is this time consuming and impractical in the long run, but it does not always work. In our last attempt to run our simulation we had arranged for a cross-site co-scheduled reservation and had submitted our job in time to start and finish during the prearranged reserved time, but our job was perpetually stuck in one of the TeraGrid site's queue.

5. Lessons Learned and Future Work

Perhaps the most important observation we can make is that Grid software and facilities, particularly for cross-site Grid applications, are not yet ready for production-level use. While the existence of Grid middleware such as the Globus Toolkit and MPICH-G2 and Grid facilities such as the TeraGrid represent significant milestones on the path to everyday Grid computing, they alone are not sufficient. Heroic effort is still needed, requiring the help of Grid middleware experts and Grid facility system administrators to run cross-site Grid applications.

Running cross-site Grid applications is significantly more complicated than running single-site applications, and therefore the error reporting by existing Grid middleware needs to be significantly improved. Also, while cross-site applications are not the only class of Grid applications found on the TeraGrid (parameter studies and functional pipelines are two other examples), they do push the facilities to their limits in order to solve problems that are otherwise impossible to solve on even the largest single computers. They are therefore an important application class, and they have unique requirements—for example, cross-site communication and co-scheduling—that must be addressed. Clearly, much work still needs to be done on the TeraGrid and other Grids to address those requirements.

6. Summary

Events like the *i*Grid 2005 conference push technology forward by providing infrastructure that is not otherwise commonly available. Individuals and groups can then use that infrastructure to give others a glimpse of what will be possible in the future. In this paper we have described such a system, one that is producing science that—until now—has not been possible at this scale or in this timeframe.

Acknowledgments

The National Science Foundation, the Office of Naval Research, and the Defense Advanced Research Projects Agency, has supported the work at Brown University. The work at the University of Chicago was partially funded by the National Science Foundation under contract SCI-0451491. The work at Argonne was supported in part by the U.S. Department of Energy, under Contract W-31-109-Eng-38. The work at Northern Illinois University was supported by the National Science Foundation, under Contract OCI-0330664. We thank the members of TeraGrid for their help in making the project a success and Pete Schmitt for his effort this pass summer at Argonne National Laboratory on the visualization portion of the project, in particular his development of the model used for the 1D visualization. We also thank the TeraGrid sites for providing computer time for this effort.

References

1. S. Dong, G. E. Karniadakis and N. T Karonis, "Cross-Site Computations on the TeraGrid," *Computing in Science & Engineering*, 7(5), 14-23, 2005.
2. I. Foster and C. Kesselman, eds. *The Grid: Blueprint for a Future Computing Infrastructure*. Morgan Kaufmann Publishers, San Francisco, 1999.
3. G. E. Karniadakis and S. J. Sherwin, *Spectral/HP Element Methods for CFD*. Oxford University Press, Oxford, UK, 1999.
4. N. T. Karonis, B. Toonen, and I. Foster, "MPICH-G2: A Grid-Enabled Implementation of the Message Passing Interface," *Journal of Parallel and Distributed Computing*, 63(5), 551-563, 2003.
5. N. Karonis, M. E. Papka, J. Binns, J. Bresnahan, J. Insley, D. Jones, and J. Link, "High-Resolution Remote Rendering of Large Datasets in a Collaborative Environment," *Future Generation Computer Systems*, 19(6), 909-917, 2003.
6. G. von Laszewski, J.A. Insley, I. Foster, J. Bresnahan, C. Kesselman, M. Su, M. Thiebaux, M. L. Rivers, S. Wang, B. Tieman, and I. McNulty, "Real-time Analysis, Visualization and Steering of Microtomography Experiments at Photon Sources," *Ninth SIAM Conference on Parallel Processing for Scientific Computing*, April, 1999.
7. S. J. Sherwin, L. Formaggia, J. Peiro, and V. Franke, "Computational Modeling of 1D Blood Flow with Variable Mechanical Properties in the Human Arterial System," *International Journal for Numerical Methods in Fluids*, 43, 673-700, 2003.

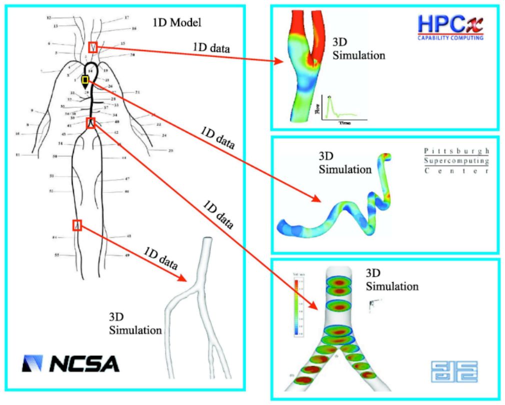


Figure 1: (in color) Artery simulation paradigms on the TeraGrid: 1D model computed on one TeraGrid site while detailed 3D bifurcation simulations are conducted on different TeraGrid sites. 1D results feed 3D simulations as inflow conditions.

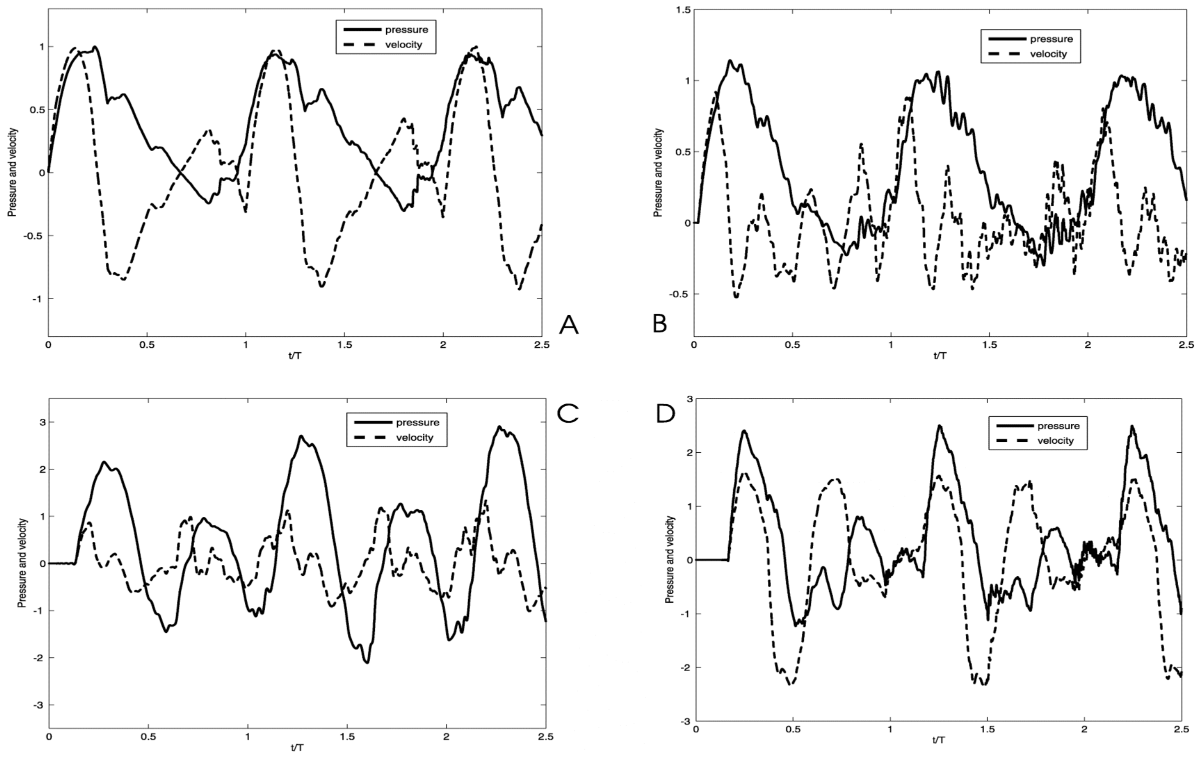


Figure 2: Time histories of velocity (dashed line) and pressure (solid line) waveforms in four arteries: (A) ascending aorta (artery 1), (B) right carotid (artery 5), (C) right ulnar (artery 9), and (D) right femoral (artery 52). Inflow condition from the heart is modeled as a half sinusoidal wave.

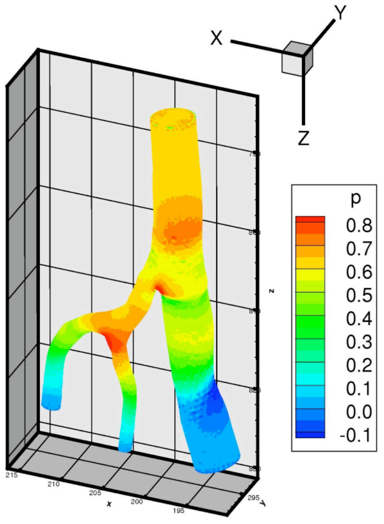
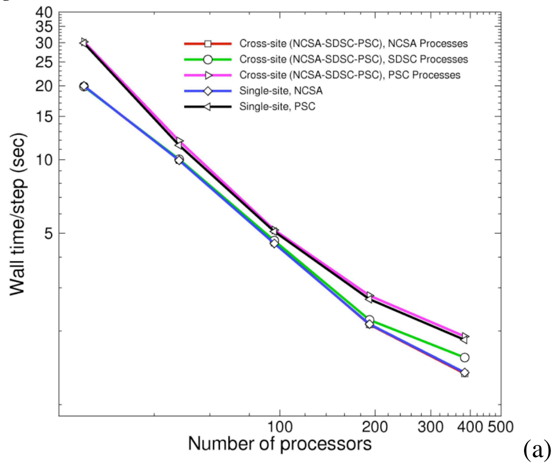


Figure 3: (in color) Isocontours of normalized pressure in the femoral-genecular arterial bifurcation in the right leg.



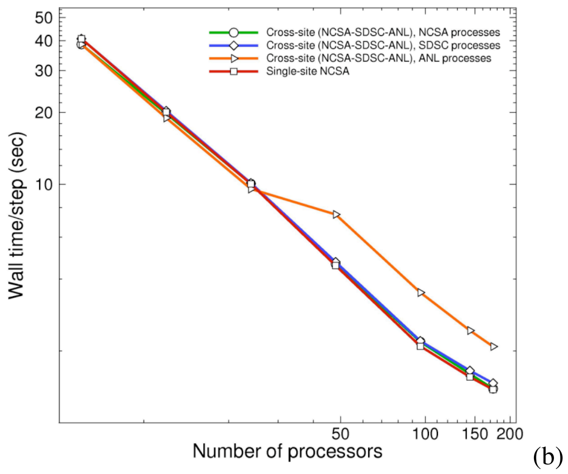


Figure 4: (in color) Performance of artery tree application with NekTar and MPICH-G2: (a) fixed problem size (a total of 6 arteries) in NCSA-SDSC-PSC cross-site and NCSA and PSC single-site runs; (b) fixed problem size (a total of 3 arteries) in cross-site NCSA-SDSC-UC/ANL cross-site and NCSA single-site runs. In cross-site runs, one-third of CPUs are from each site. Shown is wall time/step as a function of total number of CPUs.

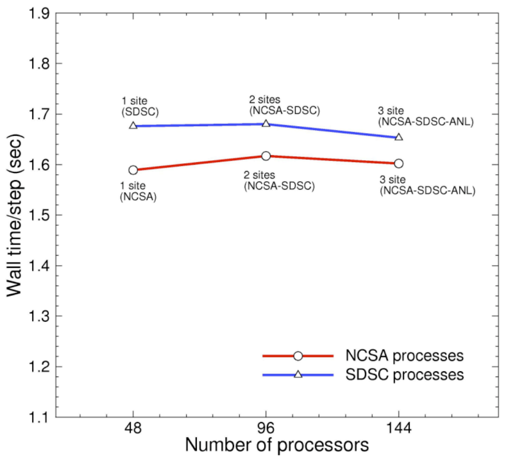


Figure 5: (in color) Fixed workload per TeraGrid site. As the number of arteries increases in simulations, the number of TeraGrid sites increases proportionally such that the workload per site remains unchanged.