From Grid to Healthgrid
T. Solomonides et al. (Eds.)
IOS Press, 2005
© 2005 The authors. All rights reserved.

A Grid Architecture for Medical Applications

Anca BUCUR[a], René KOOTSTRA[a] and Robert G. BELLEMAN[b]
[a]*Philips Research, Prof. Holstlaan 4, 5656 AA Eindhoven, the Netherlands*
{anca.bucu,rene.kootstra}@philips.com
[b]*Universiteit van Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, the Netherlands*
robbel@science.uva.nl

Abstract. Grid technology can provide medical organisations with powerful tools through which they can gain coordinated access to computational resources that hitherto where inaccessible to them. This paper discusses how several classes of medical applications could benefit from the use of Grid technology. We concentrate on applications that were put forward by partners in the Dutch VL-e project. After describing the difficulties related to the realization of such applications without making use of the Grid, we describe an architecture that allows the applications to use Grid resources. We demonstrate how this architecture can be integrated into existing systems to provide flexible and transparent access to Grid services and show performance results of a test case.

Keywords. Grid-computing, adaptive architecture, compute-intensive medical applications, fiber tracking, decomposition, speedup

Introduction

The "Virtual Laboratory for eScience" (VL-e[1]) project was initiated after the successful "Grid-enabled Virtual Laboratory Amsterdam" (VLAM-G) project [1]. In the VLAM-G project, an experimentation environment was built that allows scientists to construct experiments that are composed of geographically distributed computational resources as if they are part of the same organisation. The VL-e project will address challenging problems, including the manipulation of large scientific datasets, computationally demanding data analysis, access to remote scientific instruments, collaboration, and data dissemination across multiple organizations. The methods, techniques, and tools developed within the VL-e project are targeted for use in many scientific and industrial applications. The project will develop the infrastructure needed to support these and other related e-Science applications, with the aim to scale up and validate the developed methodology.

The VL-e philosophy is that any Problem Solving Environment (PSE) based on the Grid-enabled Virtual Laboratory will be able to perform complex and multi-disciplinary experiments, while taking advantage of distributed resources and generic methodologies and tools. In the VL-e project, several PSEs will be developed in parallel, each applied to a different scientific area. One of these areas is Medical Diagnosis and Imaging which is the focus of this research.

[1] http://www.vl-e.nl/.

This paper discusses how several classes of medical applications can benefit from parallel computing through the use of Grid technology and describes an adaptive Grid-based architecture suitable for all these application classes.

In Section 1 we introduce three computationally challenging medical applications relevant in the context of the VL-e project and perform an algorithmic analysis on each of them. In Section 2 we discuss three decomposition patterns that can be used to parallelize the applications described in Section 1. Section 3 proposes a flexible and generic architecture suitable for applications that fit the three decomposition patterns, and in Section 4 we describe a first case study of a medical application. We perform experiments in a Grid environment with a parallel version of the application and assess its performance and scalability.

1. Analysis of Medical Applications

The partners in the VL-e project have selected a number of computationally challenging applications. In particular, the clinical use of these applications is hampered by insufficient computational power. In this section we describe these applications and assess whether the underlying algorithms are suitable for parallelization.

1.1. White Matter Fiber Tractography

Recent advances in Magnetic Resonance (MR) research have opened up opportunities for gathering functional information about the brain. In addition, the development of Diffusion Tensor Imaging (DTI) offers the possibility to go beyond anatomical imaging and study tissue structure at a microscopic level *in vivo* [2]. The method uses a medical imaging technique known as Diffusion Weighted Magnetic Resonance Imaging (DW-MRI) to measure the restricted movement of water molecules along the direction of fibers. From these measurements, a tensor is constructed that describes diffusion in multiple directions. An important application of DTI is fiber tracking (FT). This application uses the anisotropic diffusion of water molecules in the brain to visualize the white-matter tracts and the connecting pathways between brain structures. Combined with functional MRI, the information about white-matter tracts reveals important information about neuro-cognitive networks and may improve the understanding of brain function. Several studies have demonstrated the feasibility of reasonably fast FT in the human brain [3]. Other research concentrates on improving the accuracy, the robustness and the throughput of the FT [4]. There are several clinical applications where FT is relevant, such as psychiatry [5], surgical planning or stroke detection.

Algorithmic Analysis: There are various solutions to FT, but the common feature is that, starting from various points, white matter fibers have to be tracked in the entire data domain. The number of detected fibers (and therefore the accuracy of the algorithm) grows with the number of considered starting points. For areas with a high concentration of fibers, too many detected fibers may lead to an indistinguishable image, which makes selection necessary. Since choosing fewer starting points would not be a good option (it would decrease the accuracy), the selection is in general performed after the fibers are generated, by specifying a number of regions of interest that the fibers have to cross. The execution time of the application depends on the number of starting points, the algorithm, and the size of the data set, and can amount to many hours. FT

would become clinically relevant if the throughput was increased without decreasing the accuracy of the result. This can be achieved by parallelization.

In order to pay off, the parallel solution has to be scalable and the amount of communication among the processors performing the algorithm has to be kept to a minimum. Fibers are tracked in the entire domain, so directly decomposing the data domain among the participating processors would not be viable due to the high need for communication and synchronization among processors. The starting points however can be distributed among processors with little extra synchronization and communication. Therefore, this problem is suited for *computational decomposition*, meaning that each processor that takes part in the computation receives the entire data domain, but the computation domain (i.e., the starting points) is divided among processors.

1.2. Functional Bowel Imaging and MR Virtual Colonoscopy

It takes many years before colon polyps become cancerous. Therefore, periodical screening can help in preventing colon cancer. However, current methods to detect colon polyps are very intrusive and are unsuitable for screening. Alternatives exist that use Computed Tomography (CT) to image the colon; the images can then be used to perform a virtual colonoscopy through scientific visualization techniques [6]. However, the ionizing radiation used in CT also makes this alternative unsuitable for screening purposes. Another alternative is to optimize the imaging strategies to exploit the higher signal-to-noise ratio of 3 Tesla MRI. In video mode, the MR scanner can generate real-time images. In combination with high-resolution static anatomical images the data will be processed and then viewed in an interactive 3D representation.

Algorithmic Analysis: In these applications the volume reconstruction for visualization is performed using a combination of image processing and isosurface extraction algorithms which construct an isosurface from a 3D field of values. The idea of the algorithm is to divide the three-dimensional geometrical space of the problem into a grid of cubes. For each cube that crosses the surface, the part of the surface contained by the cube is approximated by the appropriate set of polygons. The union of all polygons approximates the surface and the precision of the approximation is proportional with the grid resolution.

The basic approach is to traverse all cells in the space to find the approximation of the surface in each cell. For good accuracy of the result, the data set needs to be quite large, which can lead to long execution times. This problem can be parallelized by splitting the data domain into a number of sub-domains which are then distributed among the available processors. This method is called *domain decomposition*; each processor independently performs the algorithm on its sub-domain. At the end, each processor contributes to the final result with the surface generated in its own sub-domain.

1.3. Computer Aided Diagnosis and Surgical Planning in Cardiovascular Disease

Vascular disorders in general fall into two categories: Stenosis, a constriction or narrowing of the artery by the build-up over time of fat, cholesterol and other substances in the vascular wall, and aneurysm, a ballooning-out of the wall of an artery, vein or the heart due to weakening of the wall. A vascular disorder can be detected by several imaging techniques such as X-ray angiography, MRI or CT.

A surgeon may decide on different treatments in different circumstances, but all these treatments aim to improve the blood flow of the affected area. The purpose of vascular reconstruction is to redirect and augment blood flow, or perhaps repair a weakened or aneurysmal vessel through a surgical procedure. Pre-operative surgical planning would allow the a priori evaluation of different procedures under various physiologic states, such as rest and exercise, thereby increasing the chances of a positive outcome for the patient [7,8].

Algorithmic Analysis: The current approach for the treatment of patients with vascular disease is to first image the diseased area through angiography, then to plan the treatment, possibly followed by surgical intervention. We envision a simulated vascular reconstruction environment to provide decision support information to a vascular surgeon during treatment planning. To achieve this, our proposed system consists of the following: First, the vascular geometry of the patient is acquired through volumetric scanning methods (such as CTA or MRA). This scan needs to be of sufficient resolution and contrast to allow accurate isolation of the vascular morphology from the scan, including the diseased area. Next, image segmentation is used to isolate the vascular morphology from the scan. The resulting morphology is used to construct a computational data structure that can be used in a blood flow simulator, which is used to simulate the physical properties of blood flow through the vascular geometry. Properties that are of interest include pressure, flow velocity and wall shear stress. The results of this simulation are presented to the surgeon through scientific visualization methods. Based on the visualization, the surgeon will be able to propose a viable treatment. The surgeon simulates the treatment by interactively altering the computational data structure used in the flow simulation. Based on this new data structure, the flow simulator calculates a new flow solution and presents the results to the surgeon.

The system just described consists of several tightly interfaced software components [7]. Each of these components has its own unique computational requirements. Therefore, they can execute independently from each other in parallel. This decomposition method is called *functional decomposition*.

2. Decomposition Patterns

As described in Section 1, the applications that we have presented fit into three classes of decomposition patterns which allow them to exploit parallelism. Their algorithms exhibit a significant degree of spatial locality in the way they access memory as well as time locality in the sequence of operations that are performed.

It may be expected that the problem sizes will increase in time, requiring increasingly powerful computational resources. Furthermore, with the availability of increasing computational power and wide access to (geographically) distributed resources it can be expected that new applications will emerge and that some of the existing ones will gain importance. In this context, our goal is to provide a general architecture that is suitable for a wide range of applications. Besides improving the performance of each application, the architecture should be scalable relative to the data volume, and should allow changes in the computational algorithm with a minimum of changes in the algorithmic structure and no change at all in the architecture itself. By studying the common and differentiating features of the application classes that we selected, we are able to design an adaptive Grid architecture which can be applied to applications fitting at least one of the decomposition patterns.

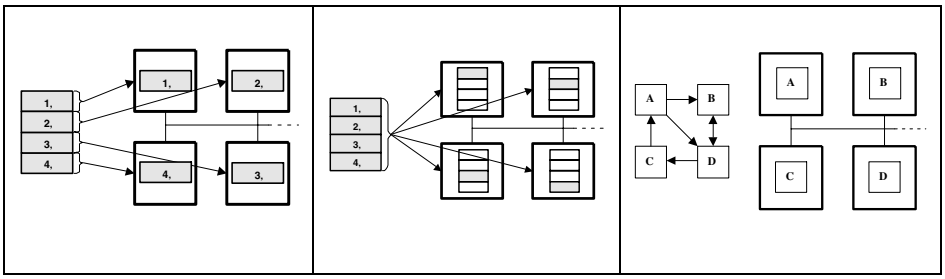

Figure 1. Decomposition patterns; (a) domain decomposition, (b) computational decomposition, (c) functional decomposition.

2.1. Domain Decomposition

With this decomposition pattern, the data domain of the application is split into disjoint partitions among the participating processors. Each processor performs the same algorithm on its own partition of data, preferably with a minimum amount of communication or synchronization (see Figure 1(a)). When there is no communication among processors we speak of pure domain decomposition. Examples of algorithms in this group are the image processing and isosurface extraction algorithms used for the purpose of volume reconstruction as described in the previous section. When the processors need to exchange data during the execution of the application we speak of domain decomposition with data exchange. The communication may occur at few isolated instances or may have an iterative nature. In this class of applications are various image processing, scientific visualization and computational simulation algorithms.

2.2. Computational Decomposition

With computational decomposition, each processor performs the same set of computations on a disjoint part of the domain but needs access to the entire data set (see Figure 1(b)). The computational domain of the application is split among the processors, while the data domain is shared. This paradigm applies to the FT application described in the previous section. This application will be described in more detail in Section 4.

2.3. Functional Decomposition

For this decomposition pattern it is characteristic that several algorithms are performed on several data sets in fixed succession. It is in fact a specialization of activities among processors: Each processor is responsible for the execution of one algorithm, then the data set is passed to another processor for performing the next algorithm (see Figure 1(c)). The current processor then moves on to the next data set, resulting in a pipelined execution. An example of an application that fits this paradigm is the vascular reconstruction application described in Section 1; here image processing algorithms provide input to a flow simulation algorithm, which in turn provides input to a scientific vvisualization algorithm.

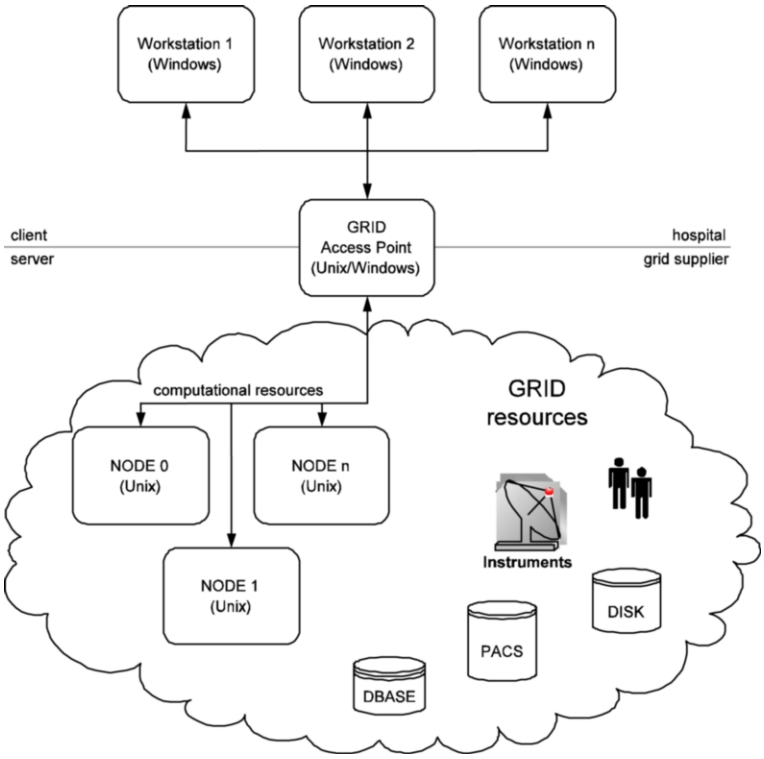

Figure 2. General architecture for solving compute-intensive medical applications using Grid technology.

3. GAMA: Grid Architecture for Medical Applications

The three decomposition patterns described in the previous section can all be modeled within a general framework. Figure 2 depicts our adaptive architecture designed to simultaneously support several applications fitting at least one of the decomposition patterns. As this figure shows, we chose for the client-server architecture; the server can simultaneously provide different sets of services for each of the application clients. Our primary intention is to make this framework minimally invasive, in the sense that the influence on the end-user is as small as possible. Instead of being enforced, the Grid-based solution will either be offered as an option to the user, or the client application will automatically choose whether to use external Grid resources or not. In the event that insufficient resources are available, the applications should automatically fall back to their local version.

Currently, the applications (situated in the hospital) run on Windows-based workstations. At the other end, Grid technology is centered around Globus, a software interface that provides the Grid "middleware" [9]. Globus is based on the Unix operating system. Therefore, in order to enable such applications to use the Grid for their execution, the compute-intensive part of the application has to be removed from the rest of the application and placed in the Grid environment. To provide an interface from the Windows environment to a Grid infrastructure that is designed around Globus, a Linux machine

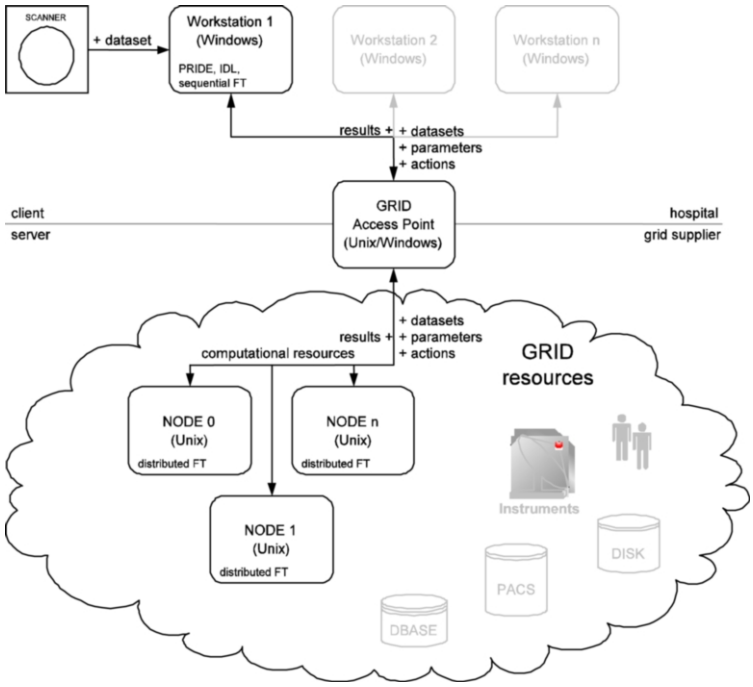

Figure 3. Applying the GAMA architecture to the fiber tracking application.

called the "Grid Access Point" (GAP) receives the requests from the hospital side, which we call the client side, and allocates the processors on the Grid, passing on the requests and returning the results to the client. The GAP may use Globus for submitting the requests to the Grid nodes, thereby exploiting the security and execution facilities offered by it. Globus is entirely Unix based, so it could not be used if we chose for directly connecting the client side to the Grid nodes. The overall performance of the application benefits from keeping the client side as little involved in the computational algorithm as possible and from restricting the communication with the rest of the system to job-submission requests only. This is because in most cases the client side will be connected to the rest of the system via slow network links. For high throughput, the GAP should be connected to the Grid infrastructure via a fast network.

4. Test Case: Fiber Tracking

As a first case study, we apply the GAMA architecture to the FT application (see Figure 3). The Grid-enabled FT application should gain performance by distributing its computational part, the FT algorithm, across Grid computational resources. We developed a parallel version of the FT algorithm and assessed its performance for several sets of application parameters and different numbers of processors. The purpose of this experiment is to check whether this type of application can benefit from our Grid-based architecture.

4.1. The Environment

The sequential FT application was built with the Philips Research Imaging Development Environment (PRIDE) on a Windows NT-based machine using the Interactive Data Language (IDL)[2].

The sequential FT application is based on a prototype application running on a single Windows workstation. It was modified to generate data sets, parameters and results. We ported this application to Unix and parallelized it. Our experiments with the parallel version of FT were performed on the second-generation Distributed ASCI Supercomputer (DAS-2)[3]. The DAS-2 is a wide-area computer system located at five Dutch universities. It consists of 200 nodes split into five clusters, one with 72 nodes, the other four with 32 nodes each. Programs are started on the DAS-2 using the PBS batch queuing system, which reserves the requested number of nodes for the duration of a program run. We submitted the jobs to the DAS-2 using Globus and used MPI to implement parallelism. The outputs of the executions of the sequential FT application on the single workstation and of the parallel FT application on the DAS-2 system were compared in order to verify the correctness of the distributed solution.

4.2. Results

In this section we present results of experiments performed with our parallelized version of FT. With full volume fiber tracking (FVFT), the starting points are evenly distributed in the entire domain. Compared to other solutions, e.g. placing starting points only in the regions of interest (ROIs), FVFT has higher computational needs but also higher accuracy, detecting a larger number of fibers and also detecting splitting and crossing fibers.

When comparing the performance of the sequential FT application to the distributed FT, we scaled the values to take into account the difference in CPU speed between the DAS-2 nodes and the workstation running the sequential version. However, these two applications run on different architectures and the comparison is only relevant as an indication of the potential performance gain through parallelization.

Our first experiments show that tracking long fibers takes noticeably longer than tracking short fibers, or checking areas with no fibers. It is also the case that fibers are in general grouped in large bundles. Since in our solution jobs are rigid (i.e. all tasks start and end at the same time), the longest task determines the execution time. This implies that simply splitting the computational domain into a number of sub-domains equal to the number of processors is not an efficient solution: Processors receiving parts of the domain with many long fibers perform a large amount of work, while processors receiving parts of the domain with no fibers spend most of the time waiting. As an alternative, we designed a solution which splits the domain on one of the axes in slices of width equal to the size of the voxel. These slices are then distributed among the processors using Round Robin, and that yields a better workload balance.

The fraction of the algorithm that tracks a fiber from a starting point is inherently sequential and limits the speedup. For identical data sets, we compare two cases differentiated by the step size for tracking the fibers.

---

[2] http://www.rsinc.com/idl/.

[3] http://www.cs.vu.nl/das2.

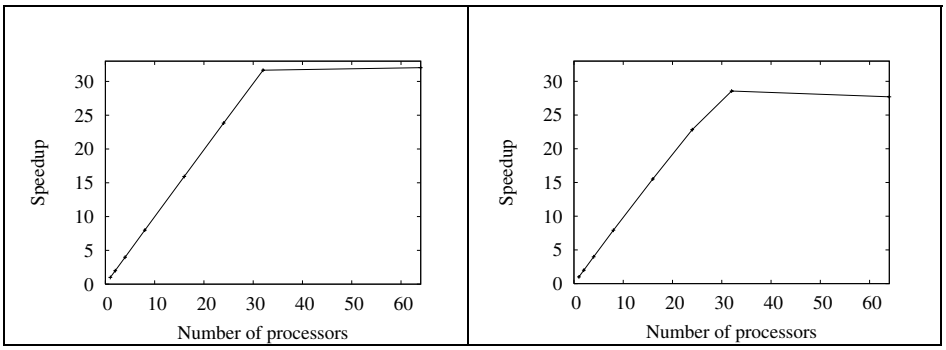

Figure 4. The speedup of the FVFT algorithm for large step (a) and small step (b) for tracking fibers.

We first studied the scalability of the application for a large step size. In this case the computation time for a single fiber is short compared to the total execution time. The execution time of the sequential FT algorithm was the equivalent of 440s, while the parallel FT algorithm took 1065s to complete on a single processor. We compared the execution time for one, two and four ROIs. The results showed that for FVFT the number of ROIs has almost no influence on performance. We ran the application on up to 64 processors and concluded that the speedup is almost linear for up to 32 processors (see Figure 4(a)).

The minimum execution time is limited by the slice of starting points requiring the longest time to compute and by the initialization time, communication time, and time required to store the results to disk. For more than 32 processors the performance improvement is very small: The computation time is still reduced by further splitting the computational domain (the limit of the inherently sequential part of the algorithm is not reached yet), but the execution time is increasingly dominated by the initialization and the communication time. The time for storing the results is constant (only one of the processors writes to the disk), while the communication time increases proportionally with the number of processors (from 0.12s for 2 processors to 5.7s for 64 processors). For 64 processors, about 20% of the execution time is spent in communication.

For the same data set, we performed experiments with a very small step size. This change significantly increased the execution time to the equivalent of more than 18 hours for the sequential version of the algorithm. Figure 4(b), shows the speedup obtained in this case. Similarly to the previous case, the scalability is very good for up to 32 processors. The communication time is not influenced by the increase in computation time but only by the number of processors performing the algorithm, so it has similar values to the previous case.

Executing the application on more than 32 processors does not seem to pay off. More investigation is needed to identify the reasons for this limitation in performance and to discern whether it would similarly affect other applications fitting computational decomposition. Since we are aiming at an adaptive framework, any solution to further improve the performance should be independent on the algorithmic details of the application, so that it could suit other applications in the same decomposition class.

5. Conclusions and Future Work

The fiber tracking test case described in the previous section illustrates an example of one of the three decomposition paradigms described in Section 2. Our future work includes the connection of the parallelized FT application through the GAP to its Windows interface, and the extension of the GAMA architecture to applications fitting the other two decomposition paradigms that we have presented.

Other applications with different decomposition characteristics will also be investigated in the future, including (but not limited to) image processing, image registration, scientific visualization and computer graphics algorithms.

If GAMA were to be used in a hospital environment today, the communication overhead from transferring images to the computing back-end may be significant enough to kill the performance gain obtained from parallelization. GAMA benefits from an architecture where data is stored "closer" (in terms of time required to communicate) to the computing back-end so that communication overhead is minimized. Such a situation could occur when hospitals store medical data on remote storage resources; having the data close to the computational resources would decrease the communication overhead that now occurs in GAMA. One such situation is currently under investigation and explores the possibility of adapting SDSC's SRB[4] so that it functions as a PACS server.

Acknowledgements

Part of this work was carried out in the context of the Virtual Laboratory for e-Science project (http://www.vl-e.nl/). This project is supported by a BSIK grant from the Dutch Ministry of Education, Culture and Science (OC&W) and is part of the ICT innovation program of the Ministry of Economic Affairs (EZ). The sequential FT application that constitutes the basis for our parallel FVFT was provided by Philips Medical Systems.

References

[1] H. Afsarmanesh, R.G. Belleman, A.S.Z. Belloum, A. Benabdelkader, J.F.J. van den Brand, G.B. Eijkel, A. Frenkel, C. Garita, D.L. Groep, R.M.A. Heeren, Z.W. Hendrikse, L.O. Hertzberger, J.A. Kaandorp, E.C. Kaletas, V. Korkhov, C.T.A.M. de Laat, P.M.A. Sloot, D. Vasunin, A. Visser, and H.H. Yakali. VLAM-G: A grid-based virtual laboratory. *Scientific Programming Journal*, 10(2):173–181, 2002.

[2] S. Mori and P.C. van Zijl. Fiber tracking: principles and strategies - a technical review. *NMR Biomed*, 15(7-8):468–480, 2002.

[3] D. Xu, S. Mori, M. Solaiyappan, P.C. van Zijl, and C. Davatzikos. A framework for callosal fiber distribution analysis. *Neuroimage*, 17(3):1131–1143, 2002.

[4] N. Kang, J. Zhang, and E.S. Carlson. Fiber tracking by simulating diffusion process with diffusion kernels in human brain with DT-MRI data. *Technical Report, Dept. of Comp.Science, Univ. of Kentuchy*, 428-05, 2005.

[5] J. Zhang, L.J. Richards, P. Yarowski, P.C. van Zijl H. Huang, and S. Mori. Three dimensional anatomical characterization of the developing mouse brain by diffusion tensor microimaging. *Neuroimage*, 20(3):1639–1648, 2003.

[6] F.M. Vos, R.E. van Gelder, I.W.O. Serlie, J. Florie, C.Y. Nio, A.S. Glas, F.H. Post, R. Truyen, F.A. Gerritsen, and J. Stoker. Three-dimensional display modes for CT colonography: conventional 3D virtual colonoscopy versus unfolded cube projection. *Radiology*, 228:878–885, 2003.

[4] The SDSC Storage Resource Broker (SRB). http://www.npaci.edu/dice/srb/.

[7] R.G. Belleman and P.M.A. Sloot. Simulated vascular reconstruction in a virtual operating theatre. In H.U. Lemke, M.W. Vannier, K. Inamura, A.G. Farman, and K.Doi, editors, *15th International Congress and Exhibition, Computer Assisted Radiology and Surgery (CARS 2001)*, number 1230 in Excerpta Medica, International Congress Series, pages 938–944, Amsterdam, the Netherlands, June 2001. Elsevier Science B.V. ISBN 0-444-50866-X.

[8] Joy P. Ku, Mary T. Draney, Frank R. Arko, W. Anthony Lee, Frandics P. Chan, Norbert J. Pelc, Christopher K. Zarins, and Charles A. Taylor. *In Vivo* validation of numerical prediction of blood flow in arterial bypass grafts. *Annals of Biomedical Engineering*, 30:743–752, 2002.

[9] Ian Foster and Carl Kesselman. Globus: A metacomputing infrastructure toolkit. *Intl. J. Supercomputer Applications*, 11(2):115–128, 1997.