

A Parallel Implementation of an Agent-Based Brain Tumor Model

by

Brian M. Skjerven

A Project

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Master of Science

in

Applied Mathematics

October 2007

APPROVED:

Professor Homer F. Walker, Major Project Advisor

Professor Suzanne L. Weekes, Associate Head of Department

Abstract

The complex growth patterns of malignant brain tumors can present challenges in developing accurate models. In particular, the computational costs associated with modeling a realistically sized tumor can be prohibitive. The use of high-performance computing (HPC) and novel mathematical techniques can help to overcome this barrier. This paper presents a parallel implementation of a model for the growth of glioma, a form of brain cancer, and discusses how HPC is being used to take a first step toward realistically sized tumor models. Also, consideration is given to the visualization process involved with large-scale computing. Finally, simulation data is presented with a focus on scaling.

Acknowledgments

The author would like to thank the following people for their support and guidance:

- Dr. Kirk Jordan, IBM
- Dr. John Wagner, IBM
- Dr. Homer Walker, WPI
- Thomas Deisboeck, M.D., Massachusetts General Hospital
- Dr. Le Zhang, Massachusetts General Hospital

Contents

1	Introduction	1
2	Model Overview	2
2.1	Macroscopic environment	2
2.2	Microscopic Environment	3
2.2.1	EGFR Network	3
2.2.2	Cell Cycle	5
2.3	Cell Phenotypes	6
3	Parallel Implementation	8
3.1	Blue Gene/L System	8
3.2	Model Implementation	9
4	Simulation Results	11
4.1	Performance	11
4.2	Imaging	12
5	Visualization	15
5.1	Deep Computing Visualization	15
6	Conclusions and Future Work	17

List of Figures

2.1	Cell Signaling Pathways	3
4.1	Visualization of Time Series Data	13
4.2	Cutplane Showing Multiple Isosurfaces.	14
4.3	Imaging and Simulation: (a) Clinical MRI (b) Cell Density (c) Cell Phenotype	14
5.1	Remote Visual Networking	16
5.2	Scalable Visual Networking	16

List of Tables

2.1	Variable Definitions and Initial Values	7
4.1	Scaling Results	12

Chapter 1

Introduction

Modeling a complex, biological system such as full-grown brain tumor is a difficult task. There are a multitude of biological processes going on at different temporal and spatial scales. In addition, problem sizes can grow enormously, requiring the use of large, parallel systems. This project is an attempt to utilize high-performance computing (HPC) to model a brain tumor of real-world size. A glioma model developed by researchers at Massachusetts General Hospital has been adapted to run on IBM's Blue Gene/L supercomputer. It is a hybrid model that incorporates continuum description of the extracellular environment in which tumor cells reside and treats tumor cells as individual agents, each with a unique genotype that is controlled by a gene regulatory pathway model.

Manipulating the data generated by a machine of Blue Gene/L's size can prove troublesome, especially when trying to visualize the data. The results of the simulations run on Blue Gene/L are presented here, and they make use of a unique visualization tool called Deep Computing Visualization (DCV), which was developed by IBM to help in dealing with large data sets. In addition to the visualization, scaling results are discussed. To advance to larger problem sizes, the model and its parallel implementation must be able to run on 4,000+ processors without issue if it is to provide novel insight into the growth of a brain tumor. Finally, the clinical application of the model and Blue Gene/L's role are discussed.

Chapter 2

Model Overview

2.1 Macroscopic environment

We begin with a three-dimensional grid with each point being assigned a value for the level of extracellular glucose (X_{14}), oxygen tension (k_{44}), and growth factor TGF_α (X_1). These variables have initial conditions that are described by the following normal distributions:

$$X_1 = T_m e^{-\left(\frac{2d}{\sigma_t}\right)^2} \quad (2.1)$$

$$X_{14} = G_a + (G_m - G_a) e^{-\left(\frac{2d}{\sigma_g}\right)^2} \quad (2.2)$$

$$k_1 = k_a + (k_m - k_a) e^{-\left(\frac{2d}{\sigma_o}\right)^2}. \quad (2.3)$$

In (2.1), T_m is the maximum concentration of TGF_α in the tumor, σ_t is a parameter that governs the dispersion of TGF_α , and d is the distance from the initial location. In (2.2) G_m is the maximum concentration of glucose in the blood and G_a is the minimum blood glucose concentration. The parameter σ_g controls the dispersion of glucose. Similarly, in (2.3) k_m is the maximum oxygen tension¹, k_a is the minimum oxygen tension, and σ_o is a parameter controlling the oxygen tension dispersion [8]. For a detailed description of the variables presented here see Table 2.1 and for coefficient values see [1].

¹Oxygen tension is the concentration of dissolved oxygen at which the partial pressure is in equilibrium with the solvent

2.2 Microscopic Environment

We now consider the model at a cellular level. Within each tumor cell there are two systems: An epidermal growth factor receptor (EGFR) gene-protein interaction network and cell cycle system. The EGFR network simulates the decision processes involved with whether a cell proliferates or migrates and the cell cycle system deals with the proliferation in more detail. Figure 2.1 shows a schematic of the EGFR and cell cycle networks [1].

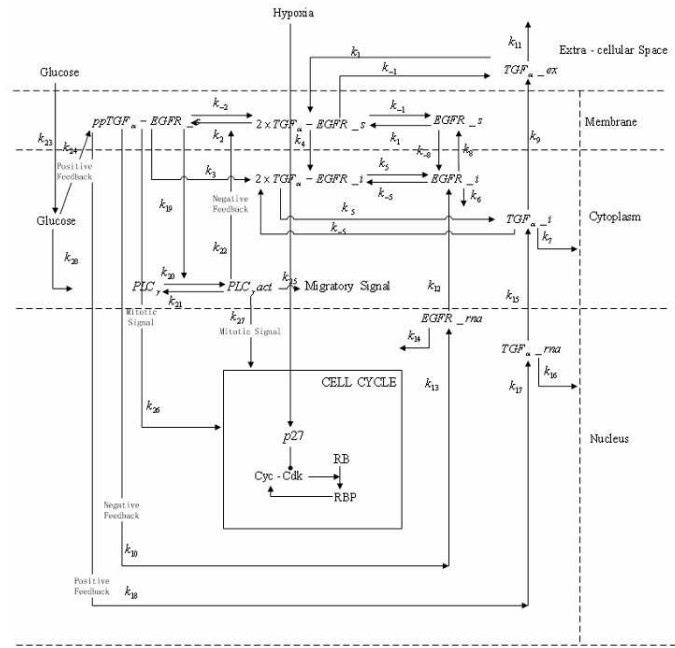


Figure 2.1: Cell Signaling Pathways

2.2.1 EGFR Network

In this model we will assume there are four layers that comprise a tumor cell and its environment. Starting outside the cell is the extracellular material, then the cell membrane, the cytoplasm, and finally the nucleus. Certain chemicals in the extracellular material can bind to receptors in the cell membrane, resulting in a reaction that impacts the internal environment of the cell. Specifically, $TGF_{\alpha(ex)}$ binds to the cell's receptor $EGFR_s$ and two $TGF_{\alpha} - EGFR_{ex}$ molecules are created

(we will refer these as $2TGF_\alpha - EGFR_s$). In addition, a phosphate group is joined to $2TGF_\alpha - EGFR_s$ (a process called *phosphorylation*), resulting in the complex $2ppTGF_\alpha - EGFR$. These reactions are modeled by the following system of ODEs [1]:

$$\frac{dX_1}{dt} = k_{-1}X_3 - k_1X_1X_2 + k_9X_7 - k_{11}X_1 \quad (2.4)$$

$$\frac{dX_2}{dt} = k_{-1}X_3 - k_1X_1X_2 + k_8X_6 - k_{-8}X_2 \quad (2.5)$$

$$\begin{aligned} \frac{dX_3}{dt} = & 2k_1X_1X_2 - 2k_{-1}X_3 - k_2X_3 [1 + w_gX_{13}] \\ & - k_3X_3 + k_{-2}X_4 + \frac{V_{M2}X_{11}}{K_{M2} + X_{11}}X_4 \end{aligned} \quad (2.6)$$

$$\begin{aligned} \frac{dX_4}{dt} = & k_2X_3 [1 + w_gX_{13}] - k_{-2}X_1X_4 \\ & - k_4X_4 - \frac{V_{M2}X_{11}}{K_{M2} + X_{11}}X_4. \end{aligned} \quad (2.7)$$

The variables X_1 , X_2 , X_3 , and X_4 denote the molecules $TGF_{\alpha(ex)}$, $EGFR_s$, $2TGF_\alpha - EGFR_s$, and $2ppTGF_\alpha - EGFR_s$. The cell then internalizes $2TGF_\alpha - EGFR_s$, resulting in cytoplasmic $TGF_\alpha - EGFR$ (X_5) which is then further broken down into cytoplasmic TGF_α (X_6) and $EGFR$ (X_7). The cell may transcribe and translate TGF_α RNA (X_8) and $EGFR$ RNA (X_9), and these RNA species, as well as the protein involved (X_{12}), may be degraded.

The phosphorylated $TGF_\alpha - EGFR$ complex can increase the rate at which inactive PLC_γ (X_{10}) is converted to active PLC_γ (X_{11}) which, in turn, inhibits extracellular glucose (X_{14}). Finally, the intake of extracellular glucose increases the level of intracellular glucose (X_{13}), which may also be reduced by $TGF_\alpha - EGFR$ phosphorylation or cell metabolism. Each of the processes is represented by the following system of ODEs:

$$\frac{dX_5}{dt} = k_3X_3 + k_4X_4 + 2k_{-5}X_6X_7 - 2k_5X_5 \quad (2.8)$$

$$\frac{dX_6}{dt} = k_5X_5 - k_{-5}X_6X_7 - k_8X_6 + k_{-8}X_2 + k_{12}X_8 - k_6X_6 \quad (2.9)$$

$$\frac{dX_7}{dt} = k_5X_5 - k_{-5}X_6X_7 - k_9X_7 + k_{15}X_9 + k_7X_7 \quad (2.10)$$

$$\frac{dX_8}{dt} = k_{13}X_{12} - k_{14}X_8 \quad (2.11)$$

$$\frac{dX_9}{dt} = k_{17}X_{12} - k_{16}X_9 + k_{18}X_4 \quad (2.12)$$

$$\frac{dX_{10}}{dt} = k_{21}X_{11} - k_{20}(k_{29} - X_{11})X_4 \quad (2.13)$$

$$\frac{dX_{11}}{dt} = k_{20}(k_{29} - X_{11})X_4 - k_{21}X_{11} \quad (2.14)$$

$$\frac{dX_{12}}{dt} = k_{16}X_9 - k_{14}X_8 - k_{13}X_{12} - k_{17}X_{12} \quad (2.15)$$

$$\frac{dX_{13}}{dt} = k_{23}X_{14} - k_2X_3X_{13} - k_{28}X_{13}. \quad (2.16)$$

2.2.2 Cell Cycle

We now turn to the interior cell dynamics and examine the process involved with determining if a cell divides. The central element here is a biological “on-off” switch which is triggered by the mass of the cell (X_{17}). This switch is influenced by cdh1-APC complexes (X_{15}) and cyclin-CDK (X_{16}); a cell proliferates when $X_{15} < k_{30}$ and $X_{16} > k_{31}$, where k_{30} and k_{31} denote the cdh1-APC and cyclin-CDK thresholds, respectively. Also note that the effect of CDK can be inhibited by non-phosphorylated retinoblastoma protein (X_{19}), or RBNP [8]. The cell cycle is also affected by the amount of oxygen and protein p27 (X_{18}) present. If oxygen levels are high there will be less protein p27 and, thus, a shorter cycle. Conversely, under hypoxic conditions there will be more protein p27 and the cell cycle will be inhibited. Lastly, the mass of the cell influences the amount of protein p27. The entire cell cycle is described by the equations,

$$\frac{dX_{15}}{dt} = \frac{(1 + k_{32}X_{19})(1 - X_{15})}{k_{34} + 1 - X_{15}} - \frac{k_{33}X_{15}X_{16}X_{17}}{k_{35} + X_{15}} \quad (2.17)$$

$$\frac{dX_{16}}{dt} = k_{39} - (k_{36} + k_{37}X_{15} + k_{38}X_{18})X_{16} \quad (2.18)$$

$$\frac{dX_{17}}{dt} = k_{40}X_{17} \left(1 - \frac{X_{17}}{k_{41}}\right) \quad (2.19)$$

$$\frac{dX_{18}}{dt} = k_{42} \left(1 - \frac{X_{17}}{k_{41}}\right) - \left(k_{43} \frac{k_{44}}{k_{44} + k_{45}} X_{18}\right) \quad (2.20)$$

$$\frac{dX_{19}}{dt} = k_{17}X_{12} - k_{16}X_9 + k_{18}X_4. \quad (2.21)$$

2.3 Cell Phenotypes

In this model we will assume that a tumor may enter one of four states: migration, proliferation, quiescence, or apoptosis. We note that cells may proliferate or migrate, but not simultaneously, and also each state is reversible except for apoptosis. A cell will decide to migrate based on the level of PLC_γ , which we will denote as the threshold σ_{PLC} . If $\sigma_{PLC} < \left|X_{11}^{t_{i+1}} - X_{11}^{t_i}\right|$, where superscripts denote time steps, the cell will enter a migratory state. Otherwise, the cell will proliferate or stay quiescent. If a cell chooses to migrate it will evaluate its neighboring locations and the determine the ideal one. This evaluation is described by the following:

$$T_j = \psi L_j + (1 - \psi)\epsilon_j \quad (2.22)$$

In (2.22), T_j is the perceived attractiveness of location j , L_j is correct evaluation of that location, $\epsilon_j \sim N(u, \sigma^2)$ is an error term, and $0 \leq \psi \leq 1$ is the search precision [4]. The search precision governs how accurate the cell is in searching for a location. If $\psi = 1$, then there is no error in how the cell perceives that location. However, if $\psi = 0$ the tumor cell exhibits random-walk motion when searching for a location.

If a cell chooses to proliferate then it will produce two identical daughter cells. These cells begin in an “undecided” state and proceed to evaluate nutrient levels in order to determine if they will become proliferative, quiescent, migratory, or apoptotic. If a cell is unable to find an unoccupied location for migration or proliferation it will become quiescent. Also, a cell may become quiescent if $\sigma_{PLC} \geq \left|X_{11}^{t_{i+1}} - X_{11}^{t_i}\right|$

or if $0.8 \leq X_{14} \leq 1.6$. Finally, if the glucose level is too low, i.e., $X_{14} < 0.8$, the cell will become apoptotic.

Table 2.1: Variable Definitions and Initial Values

Variable	Definition	Initial Value
X_1	TGF_α extracellular protein	1
X_2	$EGFR$ cell surface receptor	25
X_3	Dimeric $TGF_\alpha - EGFR$ cell surface complex	0
X_4	Phosphorylated active dimeric $TGF_\alpha - EGFR$ cell surface complexes	0
X_5	Cytoplasmic inactive dimeric $TGF_\alpha - EGFR$ complex	0
X_6	Cytoplasmic $EGFR$ protein	0
X_7	Cytoplasmic TGF_α protein	1
X_8	$EGFR$ RNA	1
X_9	TGF_α RNA	0
X_{10}	PLC_γ , inactive, Ca-bound	1
X_{11}	PLC_γ , active, phosphorylated, Ca-bound	0
X_{12}	Nucleotide pool	5
X_{13}	Cytoplasmic glucose	1
X_{14}	Extracellular glucose	0
X_{15}	cdh1-APC complex	0.9
X_{16}	cyclin-CDK	0.01
X_{17}	Tumor cell mass	5
X_{18}	Protein p27	0
X_{19}	Non-phosphorylated retinoblastoma protein	1

Chapter 3

Parallel Implementation

The computational aspects associated with modeling a brain tumor of a realistic size can be prohibitive. Current models deal with tumor cell numbers on the order of $10^5 - 10^7$, whereas realistic brain tumors can be composed of $10^{10} - 10^{12}$ cells. In addition to novel numerical methods, high-performance computers capable of reaching the teraflops and petaflops scale are needed in order to tackle such problem sizes.

3.1 Blue Gene/L System

One such system is the Blue Gene/L (BG/L) architecture that was developed jointly by IBM and Lawrence Livermore National Laboratory. Blue Gene/L is currently the world's fastest supercomputer with a LINPACK benchmark of 280.8 Tflops and a sustained 100 Tflops on a real-world application. It is a distributed memory machine designed to contain a large number of processors in a small area while minimizing power consumption.

A BG/L compute node contains 512MB of memory, dual 700MHz PowerPC CPUs, each with a double-pipeline, double-precision floating point unit, and has a peak performance of 5.6Gflops. The compute node is capable of operating in two modes: co-processor or virtual node. In co-processor mode one processor is dedicated to computation and the other is dedicated to communication, while in virtual node mode each processor handles its own communication. A full BG/L rack is composed 1024 compute nodes (or 2048 processors) with 512GB of memory and 128 I/O nodes.

In addition, BG/L has five interconnect networks for communication, I/O, and diagnostics. Of primary interest are the communication networks, which are a three-dimensional torus network, a collective network, and a barrier network. The three-dimensional torus network connects every node through a low-latency, high-bandwidth network and allows for nodes to communicate with all other nodes. The combination of BG/L's toroidal network and its fast interconnect speeds make BG/L well-suited for applications involving nearest-neighbor communications (such as the above tumor growth model). The collective network allows for any node to broadcast data to all other nodes (or any subset of nodes), as well as the global broadcasting of data. Finally, the barrier network is used for global interrupts and barriers.

3.2 Model Implementation

To implement the tumor growth model on BG/L, an existing serial C++ program was modified to incorporate the industry standard Message Passing Interface (MPI) used for processor communication. Within the program we consider both a global and a local (i.e., located on each processor) grid size. Based on the total number of processors available and a specified local grid size, the program divides the domain into smaller computational volumes. For example, suppose we perform a small run on 64 processors (4 processors in the x-direction, 4 in the y-direction, and 4 in the z-direction), each with a local size of $32 \times 32 \times 32$ grid points. Then there will be a total of $128 \times 128 \times 128$ computational volumes and each processor is responsible for 32,768 computational volumes.

The computations associated with each volume are divided into two categories: those that deal with the macroscopic environment and those related to the intracellular networks. The current model and its BG/L implementation have effectively no computation associated with the macroscopic environment. Only the initial distribution for each molecular species (TGF_α , glucose, and oxygen tension) is computed and subsequent time steps require only that the level of each species be communicated for a specific location. Regarding the intracellular networks, each computational volume can contain one or more virtual tumor cells, and for each cell the system of ODEs described in (2.4)-(2.21) must be solved. The solution is approximated using a classical second order Runge-Kutta scheme [7]. Once the local computations are completed, each virtual tumor cell undergoes the phenotype decision process

outlined in §2.3. A global sort identifies the phenotype of each virtual cell in each computational volume. Cells that have proliferated or migrated to locations outside of their original processor are then sent to the appropriate processor, and the algorithm repeats.

Chapter 4

Simulation Results

Here we present the results of several simulations conducted on Blue Gene/L, as well as several images obtained from the data set of a small run. The Blue Gene runs were conducted on both an internal IBM machine and on Harvard's School of Engineering and Applied Sciences system (SEAS). The IBM Blue Gene is a custom configuration containing only 128 compute nodes (256 processors or 1/8th of a standard Blue Gene rack). Because the SEAS system is shared among researchers at Harvard, the IBM system was used primarily for development and for small (8 and 64 processor) runs. An IBM Z Pro Intellistation with Dual 2.8 GHz Intel Xeon processors and an nVidia Quadro FX 2000 graphics card was used for rendering and also acted as the RVN server. The Amira software package was used for the visualization and a Thinkpad T60p functioned as the RVN client.

4.1 Performance

The simulations presented here were conducted on a 128x128x128 grid with a grid spacing of 10 μm in each direction, and approximately 1,000 initial virtual tumor cells. A standard forward-Euler method was used for the time-stepping algorithm. A fixed time step of 10 seconds was used for 450,000 time steps, yielding a simulated time period of approximately 52 days. A total of 19 variables were used, three of which include extracellular molecular species and the rest correspond to the cell signaling pathways. To demonstrate strong scaling the overall problem size was held fixed while the local problem sized was decreased as more compute nodes were added. The results are presented in Table 4.1. Note that in Table 4.1, one dual-processor

compute node is used for comparison and the run time is taken from [1]. Also, the simulations did not include input and output, which would have a significant impact on the overall run time; these simulations were simply to focus on the computation and to test scaling. For additional perspective, results from a $1024 \times 1024 \times 1024$ grid are included for the the largest number of compute nodes.

Compute Nodes	Run Time	Speedup
1	25 hrs, 46 min ²	N/A
8	5 hrs, 34 min	1
64	42 min	7.95
512	5 min, 21 sec	62.4
512	43 hrs, 20 min ³	N/A

Table 4.1: Scaling Results

As we can see there is almost linear scaling as the number of compute nodes is increased. The second 512 node in 4.1 run was conducted to examine the effects of an increase in the global problem size. Simple calculations suggest that if we were to use 4096 computes nodes (or 4 full Blue Gene/L racks) a simulation with a $1024 \times 1024 \times 1024$ grid size could be completed in 5 minutes, 25 seconds. The fact that this model scales fairly well indicates that a very large simulation could be completed in a reasonable amount of time. However, the issue of input and output would need to be addressed.

4.2 Imaging

Several time steps from data generated by an 8 processor run are presented in Figure 4.1. Black corresponds to necrotic cells, green for quiescent cells, red for proliferative cells, and blue for migratory cells. As the tumor grows, we can see that that heterogeneity of the tumor’s interior has given way to a large necrotic center. Figure 4.2 shows an alternate view with the interior of the tumor mass exposed and the other isosurfaces visible.

The data generated here could possible be used in a clinical environment. Standard MRI (Fig. 4.3(a)) only provides images of cell density. Also, current MRI

²Using a $64 \times 64 \times 64$ grid.

³Using a $1024 \times 1024 \times 1024$ grid.

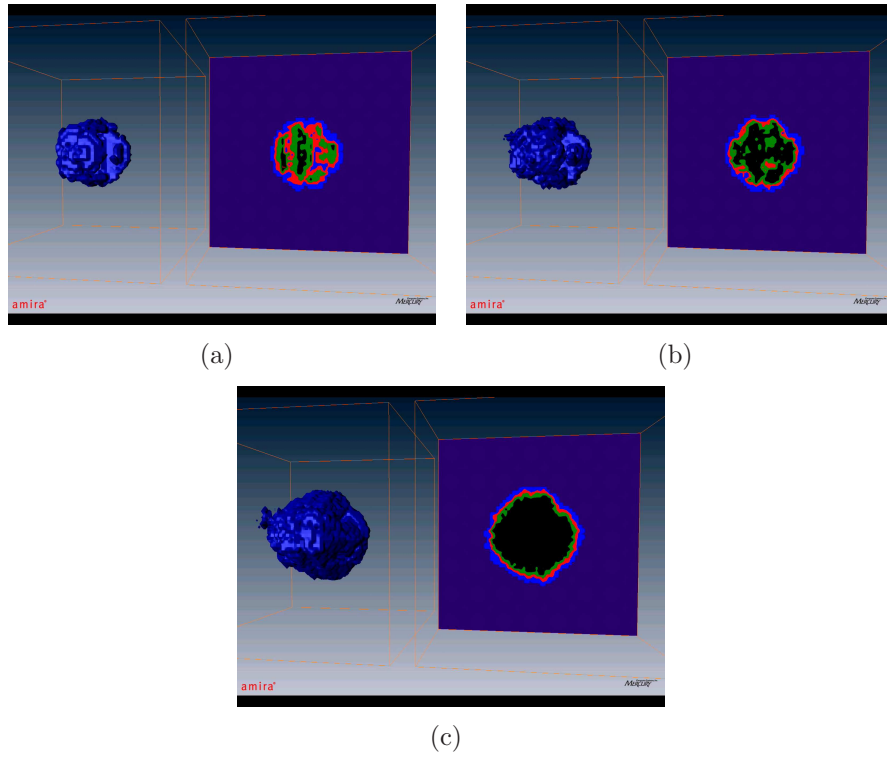


Figure 4.1: Visualization of Time Series Data

techniques don't allow clinicians to see the beginning stages of a tumor. By coupling MRIs with cell density data (Fig. 4.3(b)) and phenotypic data from simulations (Fig. 4.3(c)) it would be possible to not only detect small tumors, but also determine their makeup, thereby allowing for more specific treatments.

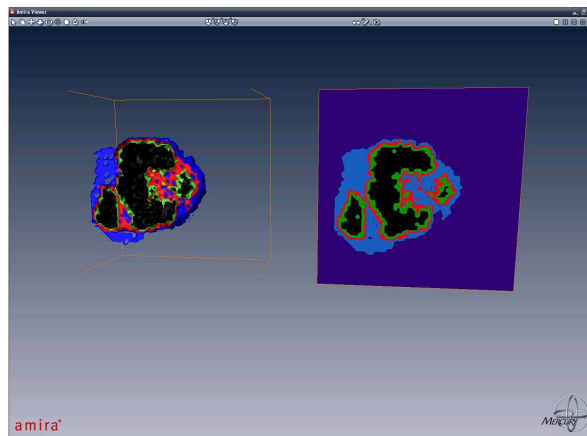


Figure 4.2: Cutplane Showing Multiple Isosurfaces.

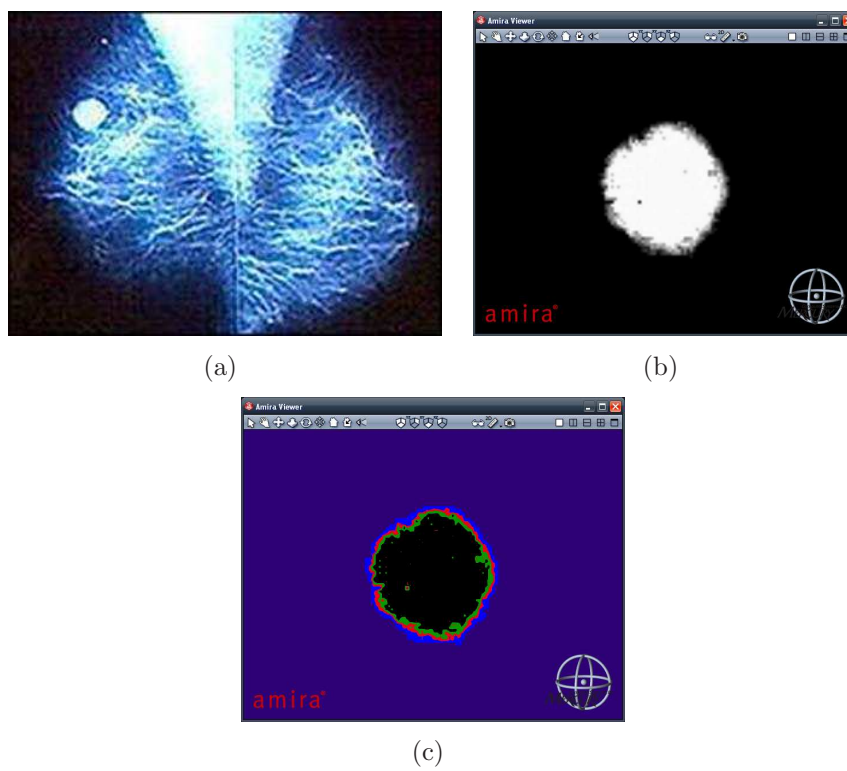


Figure 4.3: Imaging and Simulation: (a) Clinical MRI (b) Cell Density (c) Cell Phenotype

Chapter 5

Visualization

Massively parallel systems like Blue Gene/L allow researchers to tackle complex, large-scale problems. However, with larger problem sizes come larger data sets, as well as the need to visualize them in a realistic time frame. When dealing with large data sets users can be limited by software issues and insufficient hardware. In an attempt to address these problems, IBM has developed the Deep Computing Visualization (DCV) package. Deep Computing Visualization allows for existing, OpenGL-based applications to be used in either an immersive, scalable environment (Scalable Visual Networking or SVN) or for remote collaboration (Remote Visual Networking or RVN).

5.1 Deep Computing Visualization

Remote Visual Networking allows users to remotely interact with a 3D application, greatly increasing collaboration without requiring each user to have access to high-end graphics workstations. DCV works as middleware between the 3D application and the graphics processing unit (GPU), both of which are located on a server workstation. When the application makes an OpenGL call to the GPU, RVN intercepts the call and compresses the 3D data into 2D images which are then transferred, via TCP/IP, to the remote users. This process places all of the GPU-intensive calculations on the server side and gives each remote user access to a high-end graphics workstation without having to physically be at the workstation. It also reduces the collaborators' costs, as only one high-end graphics workstation is needed. In addition, virtual network computing provides users with a graphical interface to

the server workstation and also handles the transmission of mouse and keyboard commands, 2D images, and any window or menu interfaces. Figure 5.1 shows the process of RVN [5].

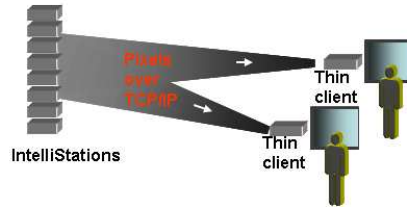


Figure 5.1: Remote Visual Networking

Scalable Visual Networking provides a method for displaying OpenGL applications on multiple, large-scale monitors. This is particularly useful when working with a large group of people or when using an immersive environment. Much like RVN, SVN intercepts the OpenGL calls but then, using MPI, distributes the 3D geometry (note that it is uncompressed) across a high-speed network to a cluster of workstations. Each workstation then renders a portion of the object, which is then displayed, producing a scaled-up version of the original image in its entirety. It should be noted that SVN and RVN may operate simultaneously. Figure 5.1 outlines SVN [5].

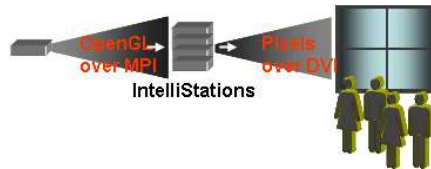


Figure 5.2: Scalable Visual Networking

Chapter 6

Conclusions and Future Work

The next step in improving the mathematical model is to incorporate diffusion of oxygen, glucose, TGF_α , and other molecular species. In addition, sources and sinks such as blood vessels need to be included. At the microscopic scale, it may be necessary to include other cellular pathways that model things like apoptosis or cell stress. Of course, the inclusion of added pathways and diffusion will impact what numerical method is utilized. An implicit time-stepping method, such as a backward-differentiation formula (BDF) method [2], may be implemented in the future; in the large-scale case of interest, this would entail using iterative linear algebra methods such as the conjugate-gradient method [3] or GMRES [6] for the linear subproblems.. Also, the addition of diffusion would introduce the possibility of using a multiscale modeling approach. The diffusible variables could be approximated on one grid size while the cellular dynamics are approximated on another grid size. Besides the grid sizing, variable time stepping could be used in the multiscale modeling approach.

A more accurate tumor model, along with improved algorithms, could yield impressive results. However, one must consider how theses will be implemented and, more importantly, optimized on a system such as Blue Gene/L. A robust implementation would take full advantage of every processor available by using a load-balancing scheme. The current tumor model performs an initial domain decomposition, but as the tumor grows this decomposition remains static, which poses an issue. At later time steps, the tumor has developed a large, necrotic center and only cells along the surface are proliferating or migrating. In terms of computation, the necrotic center corresponds to idle processors and the surface cells correspond to processors with

very large workloads. A load-balanced implementation of the tumor model would allow for the processors associated with the necrotic center to share the workload.

There are also areas in the visualization process that can be optimized. As of now, the program writes a text file at each time step that contains the location and phenotype of each virtual tumor cell, as well as data about glucose and TFG_{α} . For small runs this is an adequate solution, but even moderately sized runs can produce text files of over 1GB in size. This poses a problem when 40-50 files are generated and then must be moved and loaded into memory. A first step for the current program would be to eliminate the variables from the text files that aren't visualized. This would help to reduce the overall size, but more can be done. Systems like Blue Gene/L can produce enormous amounts of data faster than they can be moved to a file system. By eliminating the file system, the memory requirements for visualization would be drastically reduced. The visualization program would only need to load in a single time step and then discard it after it has been rendered.

The model presented here and its parallel implementation demonstrate that the modeling of realistically-sized brain tumors is in the near future. The scaling results presented §4.1 show that the current model could be scaled to run on several racks of Blue Gene/L with little modification. This is especially promising given that IBM will soon announce the next generation Blue Gene/P, capable of petascale computation. Besides the computational results, we have seen how the Deep Computing Visualization package can improve collaboration when working with HPC. However, there are a number of improvements that can be made to the mathematical model, the parallel implementation, and the visualization process in order to reach the goal of modeling realistic brain tumors.

Bibliography

- [1] T. Deisboeck A. Athale and L. Zhang. Development of a three-dimensional multiscale agent-based tumor model. *J. Theor. Biol.*, pages 96–107, 2007.
- [2] C. W. Gear. *Numerical Initial Value Problems in Ordinary Differential Equations*. Series in Automatic Computation. Prentice–Hall, Inc., Englewood Cliffs, NJ, 1971.
- [3] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, pages 409–435, 1952.
- [4] Y. Mansury and T. Deisboeck. The impact of search precision in an agent-based tumor model. *J. Theor. Biol.*, 2003.
- [5] J. Mills R. Arenburg and J. Sparlin. *IBM’s Deep Computing Visualization system class introduction*, 2005.
- [6] Y. Saad and M. H. Schultz. GMRES: a generalized minimal residual method for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7:856–869, 1986.
- [7] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer–Verlag, New York, Heidelberg, Berlin, 1983.
- [8] H.M. Byrne T. Alarcon and P.K. Maini. A mathematical model of the effects of hypoxia on the cell-cycle of normal and cancer cells. *J. Theor. Biol.*, pages 395–411, 2007.