Performance Enhancement and Equivalence Criteria for Cellular Automaton-Based Tumor Simulations

by

Jesse Farnham
Class of 2008

A thesis submitted to the faculty of Wesleyan University in partial fulfillment of the requirements for the Degree of Bachelor of Arts with Departmental Honors in the Computer Science Program
Acknowledgments

To my family, for support during the writing of this thesis, and throughout all of college.

To my academic, research, and thesis advisor, Prof. Eric Aaron, for an outstanding level of support and assistance throughout my Wesleyan career.

To Prof. Ami Radunskaya, who developed the simulation on which this work is based, for assistance and contributions throughout this work.

To Prof. Robert Lane, for ideas on the sources of simulation instability and the importance of simulating gene regulatory networks.

To Hunter Cai and Prof. Laura Grabel, for providing information on biological measurement techniques.

To Henny Admoni, for assistance with the \LaTeX{} formatting of this thesis, and for support and encouragement.

The work described in this thesis is based mainly on contributions from Prof. Eric Aaron. He developed the concept of enhancing performance by suppressing updates to steady-state areas, and he proposed the hierarchical data structure incorporating bitwise operations. He also provided the idea of equivalence criteria based on the inaccuracy of biological measurements and made many other conceptual contributions to this work.
Contents

List of Figures iv

Chapter 1. Introduction 2
   1. A Hybrid Cellular-Automaton Approach to Biological Simulation 3
   2. A Performance Enhancement to a Hybrid CA Tumor Simulation 5
   3. Criteria for Equivalence Between Simulations 8

Chapter 2. Related Work 11
   1. Cellular Automaton-Based Simulations 11
   2. Other Biological Simulation Methods 15
   3. Methods for Increasing Simulation Efficiency 16

Chapter 3. A Cellular Automaton-Based Tumor Simulation 18
   1. Nutrient Diffusion Model 18
   2. Nutrient Consumption Model 19
   3. Model of Cell Division and Death 20
   4. Model of Cell Movement 21

Chapter 4. Calibrating The Simulation to Improve Realism 22
   1. Adjusting Diffusion Parameters 22
   2. Obtaining Realistic Starting Conditions 24
   3. Limiting Reproduction of Healthy Cells 26
   4. Modifying Simulation Boundary Conditions 29

Chapter 5. Identifying Steady-State Areas 35
   1. Adding Steady-State Identification Functionality 35
   2. Locations of Steady-State Areas 37
   3. Discussion of Steady-State Locations 39

Chapter 6. Enhancing Simulation Performance by Suppressing Updates to Steady-State Areas 40
   1. Hierarchical Representation of Steady-State Areas 40
   2. Implementation of Partial Grid Updates 43
   3. Discussion of Performance Enhancement 46

Chapter 7. Criteria for Equivalence Between Simulations 50
   1. Criterion Based on Cell Populations 51
   2. Criterion Based on Measurements of Tissue Start State 53
   3. Criterion Based on Distance from Canonical Simulation 72
List of Figures

4.1 Tumor cell population before diffusion adjustments 23
4.2 Tumor cell population after diffusion adjustments 23
4.3 Starting tumor population 27
4.4 Healthy cell population without growth limiter 28
4.5 Zero-flux boundary conditions 31
4.6 Tumor cells: low nutrient concentrations at boundary 32
4.7 Tumor cells: high nutrient concentrations at boundary 33

5.1 Steady-state locations with different threshold settings 37
5.2 Steady-state locations at different timesteps 38

6.1 Example of the hierarchical data structure 43
6.2 Suppression of updates using the data structure 45

7.1 Cell population difference between modified and canonical simulations 54
7.2 Location of update suppression 55
7.3 Locations of cell population differences 56
7.4 Change in an equivalence region over time 63
7.5 Equivalence region of an outer tissue block 64
7.6 Non-equivalent tissue locations 66
7.7 Equivalence region created from reasonable result states 67
7.8 Example of translation with single element-based equivalence 69
7.9 Example of translation with block-based equivalence 70

8.1 Data structure with bitwise operations 85
8.2 Gene regulatory network model 92
Abstract

This paper describes a performance enhancement to a cellular automaton-based simulation of tumor growth. A cellular automaton is a grid-based data structure whose elements are updated according to specific rules. Elements represent small areas of simulated tissue and contain local cell population and nutrient concentration data for those areas. The simulation’s computational efficiency is enhanced by suppressing updates to steady-state tissue locations, areas with little change in nutrient concentration or cell population. This modification produces different tissue development from the original, canonical method in which all tissue areas are updated on every timestep. This paper defines several criteria by which a modified, efficient simulation can be considered equivalent to an original, canonical method; these criteria are applied to determine whether or not the performance enhancement can be considered equivalent to the original simulation.
CHAPTER 1

Introduction

Simulations of biological processes such as tumor growth have applications in medicine and biological studies. For example, simulations of tumor growth could be used to predict the effects of multiple treatment options on a particular patient’s tumor. Biological measurements of a cancer patient’s tumor and surrounding tissue could be used as starting conditions for multiple simulation runs, each modeling a different treatment strategy. The simulation outcomes could then be compared, and the most effective treatment strategy selected.

This paper presents a performance enhancement, developed by [1], to an existing simulation [11] of tumor growth and treatment. The modifications described in this paper improve computational efficiency but produce simulation results different from those of the original method. In this paper, several criteria are proposed [1] by which to decide whether or not two simulation methods can be considered equivalent; the criteria are then applied to judge equivalence between the proposed performance enhancement and the original simulation method.

Biological simulations can use a variety of computational techniques, such as cellular automata, differential equations, or regulatory network models; the simulation from [11] is a cellular automaton-based simulation. A cellular automaton...
is a grid-based data structure, originally described in [27], that represents a spatial region, with grid elements representing small, discretized areas of that region. Grid elements contain data specific to the area that they represent; in simulating the process of tumor growth, time is discretized, and a grid element's state at time \( t + 1 \) is affected by the states of neighboring elements at time \( t \). That is,

\[
S_{x,y}(t + 1) = f(S_{x,y}(t), S_{x-1,y}(t), S_{x+1,y}(t), S_{x,y-1}(t), S_{x,y+1}(t))
\]  

(1)

where \( S_{x,y}(t) \) is the state of element \((x, y)\) at time \( t \). All grid elements are updated concurrently. The simulation uses cellular automata to model nutrients and cell populations in a two-dimensional tissue slice; at each timestep, the entire tissue is updated by simulating nutrient diffusion, cell reproduction, and cell death. Updates are computationally intensive: a 700-day simulation of a 4.4-cm square tissue slice requires 1.5 hours on a 1.86 GHz processor with 2 GB RAM.

1. A Hybrid Cellular-Automaton Approach to Biological Simulation

The performance enhancement described in this paper (Chapter 6) is based on the simulation from [11], which uses a hybrid cellular automaton approach to model tumor growth. The simulation is considered \emph{hybrid} because some simulation properties are discrete and others are continuous. Specifically, time and spatial location are modeled discretely: Time passes in 24-hour steps, and the simulated tissue is divided into small subsections represented by grid elements of the cellular automaton.
Each grid element contains local nutrient concentration and cell population data for the corresponding tissue location. The data in all grid elements are updated synchronously with every timestep. Updates occur at two levels: First, the local processes of cell reproduction, cell death, and nutrient consumption are simulated at each grid element. Then, the processes of nutrient diffusion and cell movement between elements are simulated.

Local grid element updates are accomplished using ordinary differential equations (ODEs) [11]. In general, ODEs specify the change in simulation state as time progresses [9]; in the simulation from [11], ODEs are used for two purposes: to model cell reproduction and death by specifying cell population change as a function of nutrient concentrations, and to model cellular nutrient consumption by specifying nutrient concentration change as a function of cell populations.

ODEs assume homogeneous molecular concentrations and are therefore unable to account for spatial compartmentalization (the division of space into separate compartments), which can result in non-uniform concentrations. This makes purely ODE-based approaches unsuitable for simulating environments with blood vessels, multiple cells, or organelles in a single cell. For this reason, the simulation from [11] applies ODEs only to local grid elements, which are assumed to have homogeneous concentrations, rather than to the entire tissue as a whole.

Nutrient diffusion is modeled by a cellular automaton rule of the form shown in Equation 1. To update nutrient concentrations for the next timestep, each grid element is assigned the average concentration of it and its four neighboring
elements [11]. This rule conserves tissue-wide nutrient mass and causes nutrients to diffuse from high-concentration to low-concentration locations.

2. A Performance Enhancement to a Hybrid CA Tumor Simulation

In Chapter 6, a performance enhancement to the hybrid cellular automaton method from [11] is described. The enhancement improves simulation efficiency by decreasing the number of grid element updates that occur during the simulation. The enhancement works as described by [1]: Whenever a grid element is updated, local nutrient concentration change is calculated from the previous timestep to the current timestep. Locations with nutrient concentration change below a given threshold are marked for suppression of updates until the next timestep $t$ with $t = 0 \mod k$, where $k$ is a specific simulation parameter. Thus, the grid elements with stable nutrient concentrations are updated less often than other grid elements. Every $k$th timestep, updates to all locations are forced, nutrient concentration changes are computed, and the process repeats.

Suppressing updates to stable grid elements can improve computational efficiency while slightly affecting simulation results. If a tissue area $A$ with no cell population change has constant nutrient concentrations for a time interval $I = t_1..t_2$, then cell populations will remain unchanged in that interval. In the current implementation, areas like $A$ would be updated on every timestep in $I$, and no updates would actually change the state of $A$. If it were possible to identify the area $A$ and the timespan $I$ of constant nutrient concentrations, updates to $A$
during timespan $I$ could be suppressed with no effect on simulation results. However, areas like $A$ are unlikely to occur in actual simulation runs, because nutrient concentrations rarely remain constant over multiple timesteps. Therefore, instead of suppressing updates to areas with zero nutrient concentration change, updates are suppressed to areas with low nutrient concentration change. This increases simulation efficiency if the computational overhead of update suppression is not too expensive.

The above approach can increase simulation efficiency but also affects results, because nutrient concentrations may not be constant in locations where updates are suppressed, as demonstrated by the following example: Assume that nutrient concentration change in location $A$ is low but nonzero over timespan $t_i..t_j$. If $C(t_i)$ is the nutrient concentration at time $t_i$, then suppressing updates to $A$ over timespan $t_i..t_j$ would hold nutrient concentrations constant at $C(t_i)$ throughout the timespan. However, in the original simulation method, nutrient concentration in $A$ would be changing over the timespan; it would not remain at $C(t_i)$. If $C(t_k)$ is the original simulation’s nutrient concentration at timestep $t_k$ with $i \leq k \leq j$, then the difference in concentration between the original and performance-enhanced simulations at timestep $k$ is $C(t_i) - C(t_k)$. If the original simulation’s nutrient concentration change over the timespan is low, then the difference in results between the two simulation methods will be small. Thus, suppressing updates to areas with low nutrient concentration change minimizes the difference in results,
so the performance enhancement targets areas with low nutrient concentration change for suppression of updates.

To preserve simulation accuracy, updates to stable tissue areas should only be suppressed for a certain number of timesteps. The above description of the performance enhancement mentions a location $A$ and a time interval $I = t_1..t_2$ during which nutrient concentration change to area $A$ is low. Because there is no method for predicting a future simulation state, $t_2$, the end of the interval, cannot be known. In other words, once suppression of updates to a tissue location begins, the time at which nutrient concentration change will increase, and updates must therefore resume, cannot be determined. To address this problem, update suppression is restricted to a maximum allowable timespan; grid-wide updates are required every $k$th timestep. The parameter $k$ is the maximum number of timesteps between grid-wide updates; it must be tuned to balance simulation speed and accuracy. High values of $k$ allow update suppression for longer timespans, potentially speeding up the simulation but introducing unacceptable error; low values are likely to introduce less error but may provide a smaller efficiency increase.

The performance enhancement will only improve performance if stable locations (i.e., locations with low nutrient concentration change) actually exist in the simulated tissue. The experiments described in Chapter 5 show that this is the case: During the course of the simulation, some tissue locations change more slowly in nutrient concentration and cell population than others. In a simulation
3. CRITERIA FOR EQUIVALENCE BETWEEN SIMULATIONS

with a central, growing tumor, locations within and close to the tumor display more rapid nutrient concentration changes than locations far from the tumor. Eventually the tumor reaches its maximum radius and develops a necrotic core; locations within the core then change more slowly in nutrient concentration than locations close to the outer layer of living tumor cells. If the simulation is allowed to continue indefinitely, a steady state is reached in which grid-wide cell population remains constant and nutrient concentration change approaches zero; the increasing stability of the tissue would allow the performance enhancement to improve computational efficiency of the simulation.

3. Criteria for Equivalence Between Simulations

Experiments show that, when run on the same starting conditions, the performance enhancement described in Chapter 6 produces different results from the original, canonical method from [11] in which all locations are updated on every timestep. Criteria are therefore needed by which an alternate simulation can be considered equivalent to the canonical method. In Chapter 7, three such criteria are described based on [1]. They take a starting condition of interest, a canonical simulation method, and an alternate simulation method (the performance enhancement) as input; using this information, they indicate whether or not the alternate method can be considered equivalent to the canonical method.

The first equivalence criterion (Section 7.1) is based on a simple test: The alternate method is considered equivalent to the canonical method if and only if its tissue-wide cell populations are identical to those of the canonical method at
all timesteps. This criterion assumes that cell populations are the only simulation result of interest; nutrient concentrations are relevant only in their effects on cell populations.

The second criterion (Section 7.2) is based on the inherent inaccuracy of biological measurements. The starting tissue state is assumed to be a measurement of an actual biological tissue sample. Because such measurements have an associated error range, the actual state of the biological sample may not be identical to the simulation start state; there are a number of other reasonable start states based on the biological measurement. The second criterion defines reasonable start states by slightly modifying the original start state. It then runs the canonical simulation on these start states to produce a set of reasonable result states, which are plotted in three-dimensional space. Finally, the alternate simulation method is run on the original start state; it is considered equivalent to the canonical method if and only if its result state is within the convex hull of the reasonable result states produced by the original method. Section 7.2 provides more details on the implementation of this equivalence criterion.

The third equivalence criterion (Section 7.3) is also based on the inaccuracy of biological measurements. The modified simulation is considered equivalent to the canonical simulation if and only if the modified simulation’s cell population is within a certain range of the canonical population. The range is based on the
accuracy of biological measurements; simulation results that could not be distinguished by such measurements are considered equivalent. For example, if biological measurements have an accuracy of 10%, then simulations with cell population differences of less than 10% are considered equivalent.

The three criteria are used to test whether simulations utilizing the performance enhancement (Chapter 6, [1]) can be considered equivalent to the original simulation method from [11]; the results are reported in Chapter 7.
CHAPTER 2

Related Work

1. Cellular Automaton-Based Simulations

1.1. Biological Simulations. In addition to the simulation on which this paper’s work is based [11], several other cellular automaton-based tumor simulations exist. These simulations model several processes not simulated in [11], such as adaptive blood vessel networks and immune system response. These components could be added to the simulation from [11] in order to increase biological realism. Furthermore, the equivalence criteria discussed in Chapter 7 could be used to test the validity of performance enhancements to the simulations discussed in this section.

In [2], a cellular automaton-based model of tumor growth, the effects of differential blood flow and vessel adaptation are modeled. In this model, blood is treated as a non-Newtonian fluid which flows through a network of blood vessels. For example, red blood cells are distributed unevenly between the two daughter vessels in a network bifurcation; the daughter vessel with higher flow velocity receives a greater number of red blood cells. Furthermore, the blood vessel network adapts in order to best provide nutrients to the tissue. For example, a decrease in red blood cell flow rate may trigger an increase in blood vessel radius.
Another cellular automaton-based tumor simulation, described in [10, 19], models the immune system. Two types of immune cells are modeled: natural killer (NK) cells and cytotoxic T lymphocytes (CTLs). NK cells wander the tissue area until encountering a tumor cell, which they then attack. When a tumor cell is lysed in this manner, additional CTLs are recruited to the same site, where they will attack other tumor cells in the immediate neighborhood. Immune cells in tumor-free locations die or randomly move to neighboring locations.

IMMSIM [23] is a cellular automaton-based simulation of immune system response. The IMMSIM model simulates discrete events such as the destruction of an individual antigen by an immune cell, rather than taking a continuous, differential equation-based approach. As in [2, 10, 11, 19], biological components such as antigens and antibodies diffuse among grid elements. If two components are present in the same grid element, an interaction event such as antigen-antibody binding may occur. Each type of interaction has specific rules by which the probabilities of final outcomes are determined; the end result is then decided stochastically.

ParImm [3], a further development of IMMSIM, is another cellular automaton-based immune system simulator. Like IMMSIM, ParImm models cellular and molecular components of the immune system. In ParImm, cells are implemented as stochastic finite state machines, with states corresponding to possible cellular conditions, such as normal, resting, infected, or dead. Interactions involving cells may result in a change in cellular state. For example, an interaction between an
immune cell and an infected cell might change the infected cell’s state to “dead.”

The performance enhancement discussed in Chapter 6 was designed for a cellular automaton-based simulation [11]; similar enhancements could likely be applied to the above CA-based simulations. The equivalence criteria from Chapter 7, or modifications thereof, could then be used to decide whether the performance enhancements are equivalent to the original simulation methods.

1.2. Non-Biological Cellular Automaton-Based Simulations. Cellular automata have also been used in non-biological simulation applications, such as the modeling of physical processes, crowd dynamics, and wireless networks. The techniques demonstrated in this paper could possibly be adapted to enhance the performance of these non-biological simulations.

In [15], three-dimensional surface cellular automata are used to simulate the development of surface cracks on three-dimensional objects. Cellular automaton grid elements represent small areas of an object’s surface and store surface stress data for those areas. Cracks originate in grid elements with high surface stress and spread based on the stresses of surrounding grid elements; as cracks spread, they modify the stresses of nearby grid elements.

In [18], cellular automata are used to model structural failure of rocks. Automaton grid elements represent small areas of a rock’s surface and contain displacement and force data for those areas. Heterogeneous rock composition is
simulated by assigning different strengths and elasticities to different grid elements. As in [15], cracks occur between two adjacent grid elements and spread from their points of origin.

Cellular automata are used in [12] to simulate crowd evacuation in response to a fire. Two simultaneous processes are modeled: fire spreading and crowd evacuation. To model fire spreading, values representing local fire intensity are stored in each grid element, and fire spreads radially outward from its starting point. Crowd evacuation is modeled with an agent-based approach. Agents move among grid elements, following the path to the nearest exit, except when collision or fire avoidance requires deviations.

In [8], cellular automata are used to simulate the behavior of nodes in wireless sensor networks (WSNs). These nodes may be arbitrarily spread over an area of interest; they are able to independently discover their locations and establish a connected network of communication. Nodes are represented by grid elements in a two-dimensional cellular automaton. Grid elements contain data on the corresponding sensor node’s energy level, position, and state (active or standby mode). Due to the (possibly) arbitrary distribution of sensor nodes, certain nodes may be redundant; that is, they may not be necessary to ensure complete sensor coverage of the area of interest. Nodes that have two or more active neighbors are considered redundant and enter standby mode to conserve energy. Active nodes’ energy reserves are eventually depleted, leading to permanent shutdown and possibly requiring the reactivation of previously redundant nodes.
2. OTHER BIOLOGICAL SIMULATION METHODS

The above simulations are CA-based; because the performance enhancement described in Chapter 6 is designed for a CA-based simulation, it could possibly be adapted to increase the efficiency of the simulations discussed in this section. Though the simulations do not model biological processes, ideas from this paper’s discussion (Chapter 7) on biologically-motivated equivalence criteria might be adapted to define equivalence criteria for these non-biological, cellular automaton-based simulations.

2. Other Biological Simulation Methods

The simulation used in this paper [11] uses cellular automata as a means of computing and storing data. Multiple other methods exist for simulating biological systems. Although the simulation described in this paper is CA-based, it also uses methods such as differential equations. Methods not currently used, such as regulatory network modeling, could be added in order to improve biological realism.

Regulatory networks are collections of interacting proteins that allow cells to respond to changing environmental conditions. Signaling molecules bind to cell surface receptors, which chemically modify certain proteins. This initiates a cascade of chemical modification to various proteins, some of which influence gene expression. As a result, gene expression levels change, affecting cellular processes such as reproduction and metabolism. Several methods have been developed for simulating gene regulatory networks; among them are directed graphs [9] and ordinary differential equations (ODEs) [9, 25]. These methods could be added to
the simulation used in this paper [11], which currently does not explicitly simulate cellular regulatory networks.

3. Methods for Increasing Simulation Efficiency

This paper presents a performance enhancement for the deterministic cellular automaton simulation of tumor growth from [11]. The technique (described in Chapter 6) applies most directly to deterministic simulations. Other techniques, described below, have been developed for improving the efficiency of stochastic biological simulations.

The Gillespie algorithm [13] and the $\tau$-leap algorithm [14] are two performance enhancements for stochastic simulations. Stochastic systems can be represented by a stochastic master equation [9], which, given a state, returns the probability that the system is in that state. Directly solving the master equation for complex stochastic systems may be infeasible. The Gillespie algorithm is a more computationally efficient method to exactly solve stochastic master equations, but certain conditions (such as large differences among reaction timescales) can cause it to run slowly [20]. The $\tau$-leap algorithm is a more efficient modification of the Gillespie algorithm; one version, the binomial $\tau$-leap algorithm [7], may run 100–1000 times faster than the Gillespie algorithm. However, it only provides an approximate solution to the master equation. An alternative to the Gillespie algorithm is the Stochsim algorithm [21], which scales more favorably with simulations involving multiple-state molecules, such as proteins with multiple binding sites [20].
3. METHODS FOR INCREASING SIMULATION EFFICIENCY

Stochastic simulation methods are less efficient than deterministic methods because of the need for repeated random number generation. As demonstrated in [26], it is possible to improve the efficiency of stochastic simulations by dynamically switching between stochastic and deterministic methods as a simulation progresses. Biochemical reactions are best modeled stochastically, especially when molecular concentrations are low. In this situation, the random motions of molecules in solution cause fluctuations in reaction rates that are difficult to model deterministically. When molecular concentrations are high, reaction rates become effectively deterministic. In [26], stochastic methods are used when molecular concentrations are low to improve realism, and deterministic methods are used when molecular concentrations are high to improve efficiency. Techniques such as this, in which simulation methods are dynamically applied based on present conditions, could be the basis for further performance enhancements to the simulation used in this paper.
CHAPTER 3

A Cellular Automaton-Based Tumor Simulation

The work described in this paper is based on an existing hybrid cellular automaton simulation of tumor growth [11]. The simulation represents a thin, effectively two-dimensional slice of tissue with a cellular automaton data structure. Automaton grid elements represent small tissue areas and contain cell population and nutrient concentration data for those areas. To simulate the passage of time, all grid elements are updated in a two-step process: First, nutrient concentrations are updated by simulating diffusion and consumption by living cells. Secondly, cell populations are updated by simulating reproduction and death based on local nutrient concentrations in each grid element.

1. Nutrient Diffusion Model

Diffusion of two nutrients, oxygen and glucose, and one metabolic product, protons, is simulated as shown in Equation 2, taken from [11]: Each grid element receives the average nutrient concentration in the five-element neighborhood containing it and its four neighboring elements.

\[ N_{i,j}(t + 1) = \frac{N_{i,j}(t) + N_{i,j+1}(t) + N_{i+1,j}(t) + N_{i-1,j}(t) + N_{i,j-1}(t)}{5} \]  

(2)

where \( N_{i,j}(t) \) is the concentration of nutrient \( N \) in grid element \((i, j)\) at time \( t \).
Nutrients enter the tissue from simulated blood vessels and diffuse to locations with lower concentrations. Blood vessels are simulated by setting nutrient concentrations in designated grid elements to a high value. The rule shown in Equation 2 is then applied, causing nutrients to diffuse away from blood vessels into the tissue.

The rule from Equation 2 realistically models diffusion of nutrients from high-concentration to low-concentration areas. A high-concentration grid element surrounded by four low-concentration elements will have its nutrient concentration decreased, because the four low-concentration elements will lower the average neighborhood concentration. A symmetric argument shows that low-concentration elements surrounded by high-concentration elements will have their nutrient concentrations raised. Furthermore, the rule in Equation 2 can be shown to conserve tissue-wide nutrient mass. See [11] for further justification of the diffusion model.

2. Nutrient Consumption Model

The simulation from [11] models three cell types: healthy cells, tumor cells, and necrotic (dead) cells. Healthy and tumor cells consume oxygen and glucose according to differential equations reported in [11], which are applied to compute the consumption-induced changes in nutrient concentration for each grid element.

The nutrient consumption model has the following properties:

- Glucose consumption increases with glucose concentration.
- Oxygen consumption increases with oxygen concentration.
• Up to a saturation threshold, glucose consumption increases as oxygen concentration decreases.

• Oxygen consumption decreases as proton concentration increases.

This model is biologically motivated by two chemical processes used by cells to obtain energy: aerobic respiration and anaerobic respiration. Biological evidence for the model is presented in [6].

3. Model of Cell Division and Death

Cell reproduction and death are functions of local nutrient concentrations; they use an energy budget model [17] in which cells with an excess of energy reproduce, and cells with too little energy die. Differential equations reported in [11] are applied to all grid elements to compute cell population change as a function of energy produced through cellular respiration, which is a function of oxygen and glucose consumption rates. New cells are placed as in Section 3.4; dead, formerly healthy cells are discarded, and dead, formerly tumor cells become necrotic cells. The cell reproduction model is based on two assumptions [11]:

• Cells require a fixed amount $M$ of energy per timestep for maintenance functions essential to survival. Cells with less than $M$ energy die.

• A fixed amount $g$ of energy in excess of that used for essential survival functions is required for a cell to divide. Cells with more than $M$ energy use the excess to divide; each division expends some of the excess, and division continues until no excess energy remains.
4. Model of Cell Movement

New tumor cells are assigned locations using principles of cell adhesion; they are placed in a manner that minimizes their potential energy of interaction \( E \), a function of local cell population. Below a specific population \( p_0 \), cells are assumed to make no physical contact, so \( E \) is zero. At a specific higher population \( p_1 \), cells begin to make contact and adhere, and \( E \) decreases to a negative value, reaching its minimal value at the optimal cell population \( p_2 \). Above \( p_2 \), cell elasticity becomes more pronounced than adhesion, and \( E \) increases up to the maximum possible population \( p_{max} \) [11].

New tumor cells are placed in their grid element of origin, or in one of the four neighboring elements, according to the following rules. Tumor cells are placed first in grid elements with populations less than \( p_2 \), with densely populated elements receiving cells before sparsely population elements, in order to minimize total \( E \) over the local neighborhood. Once all elements in the five-element neighborhood containing the element of origin and the four adjacent elements are filled to \( p_2 \), the remaining cells are distributed randomly over the neighborhood until \( p_{max} \) is reached in all grid elements. Any remaining new tumor cells are discarded due to overcrowding.

Adhesion between normal cells is assumed to be negligible, so no potential energy function is used. New normal cells are distributed randomly among the five-element neighborhood surrounding the grid element of origin.
CHAPTER 4

Calibrating The Simulation to Improve Realism

Several changes were made to the simulation from [11] (described in Chapter 3) to improve biological realism. First, nutrient diffusion parameters were adjusted to produce more realistic tumor growth. Secondly, more realistic simulation starting conditions were obtained. Third, cell growth rules were changed to produce a more realistic distribution of healthy cells. Finally, three boundary conditions, simulation rules specific to elements at the edge of the cellular automaton grid, were investigated for realism.

1. Adjusting Diffusion Parameters

The simulation’s diffusion constants, which define the rate of nutrient diffusion, were increased to produce more rapid diffusion and more realistic tumor growth. Before the change, tumors grew slowly and took loose, non-circular forms (Figure 1). After the adjustment, tumors were circular and grew more rapidly (Figure 2).

Originally, tumors required hundreds of 24-hour timesteps to cover less than 25% of the grid area. Furthermore, tumors formed loose structures with intermingled living and necrotic cells, rather than the spherical form reported in [16] consisting of a necrotic core surrounded by an active outer cell layer.
Figure 1. Tumor cell population in a 51-by-51 grid at timestep 500, before the modifications described in this section were implemented.

Figure 2. Tumor cell population in a 54-by-54 grid at timestep 80, before the modifications described in this section were implemented.
The tumor form shown in Figure 1 suggested that tumors were spreading out from their points of origin toward outer tissue areas with higher nutrient concentrations. This indicated that nutrient diffusion was occurring slowly, preventing replenishment of nutrients that were consumed by tumor cells. To address this issue, nutrient diffusion speeds were adjusted.

Nutrient diffusion is simulated by the averaging technique in Equation 2. The number of averaging steps allowed per timestep varies by nutrient, allowing different diffusion speeds to be modeled: Increasing the number of averaging steps allows nutrients to diffuse over a larger area in one timestep.

Nutrient diffusion constants were obtained [24] and input into the simulation, increasing nutrient diffusion speeds. As a result, oxygen and glucose flux from high-concentration tissue areas to low-concentration areas near the centrally-located tumor increased, and tumor cells had more energy available for reproduction. The resulting changes in tumor growth characteristics are as shown in Figures 1 and 2. Tumors developed circular shapes, with an outer layer of actively-dividing cells enclosing an inner region of necrotic cells, as in another model of tumor growth [16]. Circular tumors expanded to fill most of the grid, rather than remaining confined to a small central area as before.

2. Obtaining Realistic Starting Conditions

Simulation starting conditions (i.e., the tissue state before running the simulation) were modified to improve realism. Nutrient concentrations in the original starting conditions were far from the steady-state values they would attain after
many simulated timesteps. This resulted in high tissue-wide nutrient concentration changes throughout the simulation, a characteristic that does not seem biologically realistic. To address this issue, new starting conditions were obtained by running a tumor-free simulation until it reached a steady state.

Experiments showed that nutrient concentrations in the original starting conditions were not in a steady state. Healthy cell populations in a tumor-free simulation using the original starting conditions quickly reach a stable population at which the rate of cellular energy production, as limited by the replenishing nutrient flux from blood vessels, produces just enough energy for survival but leaves none for reproduction. Nutrient concentration change then decreases throughout the tissue as concentrations stabilize. Nutrient concentration change throughout the simulation was so high that concentrations had not stabilized by the end of the 700-day simulation. Because cell populations stabilized early in the simulation, cell growth or death is not responsible for the delayed onset of steady-state nutrient concentrations. Thus, the excessive instability in nutrient concentrations is due to initial concentrations that are too far from steady-state.

New, steady-state starting conditions were obtained by modifying the simulation’s diffusion rules. To simulate diffusion, at most $n$ grid-wide averaging steps are run per timestep, where $n$ depends on the diffusion constant of the nutrient being simulated. All $n$ diffusion steps may not be run: If the latest averaging step changes nutrient concentrations by less than a threshold value $T$, no further steps are run, because they will lead to even smaller nutrient concentration changes.
The purpose of this rule is to improve simulation performance; for maximum realism, $T$ should be set to zero, causing all $n$ steps to be run every timestep.

To generate starting conditions closer to steady-state, $T$ was decreased by 1,000 times for oxygen and 10,000 times for glucose, and the simulation was run on the tumor-free original starting conditions for 5,000 timesteps. The decrease in $T$ increased the amount of nutrient diffusion per timestep, lowering the number of timesteps needed to reach steady-state nutrient concentrations. After 5,000 timesteps, cell populations were steady and nutrient concentration changes were close to zero. A seed population of tumor cells was then added to a single central tissue location, and the simulation was run for 60 more timesteps using the default value for $T$. The resulting tissue consisted of a large, outer stable area surrounding a small unstable area centered on the growing tumor (Figure 3). This tissue state was saved and used as the starting condition for all future simulations.

3. Limiting Reproduction of Healthy Cells

Reproduction rules for healthy cells were changed in order to obtain an even distribution of cells. Before the change, some tumor-free tissue locations had no healthy cells, a condition that does not seem biologically realistic. Changing the healthy cell reproduction rules produced a uniform tissue-wide distribution of healthy cells.

Examination of steady-state tissue from simulations run using the original rules from [11] showed some typical cell population patterns. In the absence of tumor cells, healthy cell populations were highest (10–12 cells) in grid elements
bordering blood vessels, which are horizontally and vertically spaced every seven grid elements. Healthy cell populations decreased with increasing distance from blood vessels, and some maximally-distant elements had no healthy cells. This distribution of cells did not seem biologically realistic (Figure 4). Healthy cell populations were zero in grid elements far from blood vessels because nutrient concentrations in those grid elements were lower than required for cell survival.

Two modifications were attempted in order to eliminate grid elements with zero healthy cell population. First, healthy cells were modified to require lower local nutrient concentrations for survival. This modification was unsuccessful; it caused an increase in healthy cell population near blood vessels, which resulted in higher nutrient consumption and still prevented healthy cell growth far from blood vessels.

Figure 3. Tumor population in the start state generated as described in this section. A 108-by-108 simulated tissue is shown; color represents tumor population at each grid element.
3. LIMITING REPRODUCTION OF HEALTHY CELLS

Secondly, a simple model of genetic growth control was implemented. In the original simulation from [11], healthy cell growth was only limited by nutrient concentrations; given a tissue with high enough nutrient supply, healthy cells would grow until every grid element was full. This behavior seems biologically unrealistic, as a distinguishing factor between healthy and cancer cells is that healthy cells do not grow without limit. Genetic controls for healthy cell reproduction were implemented by preventing healthy cell reproduction in grid elements with healthy cell population greater than a chosen value \( L_h \). \( L_h \) was chosen by testing different values until tissue-wide healthy cell concentrations were uniform; currently, \( L_h \).
4. Modifying Simulation Boundary Conditions

As shown in Equation 1, updates to cellular automaton grid elements depend on the element being updated, and on its four immediate neighbors. Elements at the edge of the cellular automaton (hence, at the edge of the simulated tissue) require their own update rules, because they do not have four neighboring elements. The update rules for edge and corner elements are known as boundary conditions. Three boundary conditions were investigated for biological realism and computational efficiency: periodic, zero-flux, and fixed.

4.1. Periodic Boundary Conditions. The original implementation, described in Chapter 3, used periodic boundary conditions, in which grid elements at the tissue’s edge are considered adjacent to elements at the opposite edge. Periodic boundary conditions lead to a biologically unrealistic artifact in the simulated tissue: Nutrient concentrations in the tissue center tend to be lower than around the edges. To remove the artifact, blood vessel locations were modified as described below.

Originally, blood vessels were placed in a grid-like pattern. The first blood vessel was placed in the upper left tissue corner; additional vessels were placed below and to the right every $b$ grid elements, where $b$ is a fixed value, until the tissue
boundaries were reached. Due to periodicity, the vessels at opposite tissue edges were effectively adjacent, producing higher blood vessel density around the edges than in the center. This led to higher nutrient concentrations and cell populations around the simulation edges. This artifact was removed by adjusting blood vessel distribution to produce even blood vessel density in all tissue locations, accounting for periodic topology of the tissue.

Periodic boundary conditions decrease computational efficiency when used with the performance enhancement described in Chapter 6. See Section 4.4.4 for details.

4.2. Zero-flux Boundary Conditions. Zero-flux boundary conditions are an alternative rule for applying diffusion to tissue borders. With zero-flux boundary conditions, no nutrient flux occurs across tissue borders. Implementation of zero-flux boundary conditions is more complex than periodic boundary conditions, because grid elements do not all have the same number of neighbors. Under zero-flux boundary conditions, edge elements have three neighbors, and corner elements have two neighbors (Figure 5). This requires modifications to the diffusion routine, which assumes that all grid elements have four neighbors: It must be parameterized with the number of neighbors for the grid element of interest, increasing complexity.

As with periodic boundary conditions, zero-flux boundary conditions decrease simulation efficiency when used with the performance enhancement described in
Figure 5. In zero-flux boundary conditions, central elements (A) have four neighbors (a). Edge elements (B) have three neighbors (b), and corner elements (C) have two neighbors (c). No diffusion occurs across the edge of the grid.

Chapter 6. For this reason, implementation of zero-flux boundary conditions was not completed.

4.3. Fixed Boundary Conditions. Fixed boundary conditions were the third boundary rule investigated. Fixed boundary conditions avoid the issue of diffusion across tissue boundaries by assigning fixed nutrient concentration and cell population values to edge grid elements. No modification to the diffusion routine is required; once it has been run, the simulation edges are simply reset to the appropriate values.

In order to ensure an even tissue-wide distribution of nutrients, appropriate nutrient concentrations had to be chosen for the tissue edges. To accomplish this, a tumor-free simulation was run with periodic boundary conditions until it had entered a steady state, with similar nutrient concentrations throughout the tissue.
Figure 6. Tumor cell population at day 50 with fixed boundary conditions; boundary oxygen concentrations are 10 times lower than in Figure 7. This prevents the tumor from growing as rapidly as in Figure 7.

Nutrient concentrations at the tissue’s edge were then used as the fixed boundary conditions for future simulations.

Varying the edge nutrient concentrations used in fixed boundary conditions can have a large effect on simulation results. It is therefore important to calibrate fixed boundary conditions for maximum biological realism. Figures 6 and 7 show the effects of changing edge oxygen concentrations in a simulation using fixed boundary conditions.

Fixed boundary conditions do not adversely affect computational efficiency when used with the performance enhancement from Chapter 6; see Section 4.4.4 for further details.
Figure 7. Tumor cell population at day 50 with fixed boundary conditions; boundary oxygen concentrations are 10 times greater than in Figure 6. This allows the tumor to grow more rapidly than in Figure 6.

4.4. Discussion. Realism and efficiency concerns influence which boundary conditions should be used for an individual simulation.

The biological realism of the three boundary conditions depends on the type of tissue being simulated. Periodic boundary conditions model the tissue as a closed unit; they may be most realistic when modeling a self-contained tissue, such as an organ, to or from which nutrients cannot easily diffuse. Zero-flux boundary conditions are more appropriate for two-dimensional tissue slices whose boundaries are a barrier to diffusion, such as cell cultures grown on a plate. Fixed boundary conditions are most realistic for simulation of small tissue areas embedded in a larger nutrient bath or entire organism.
Efficiency concerns also influence the choice among boundary conditions: Application of a given boundary rule may be efficient in one simulation method but inefficient in another. For example, periodic boundary conditions are the most efficient of the three rules when applied to the original method from [11]. However, when applied to the performance enhancement described by [1] (Chapter 6) periodic boundary conditions require repeated use of the modulus operator, reducing efficiency. In contrast, fixed boundary conditions are slightly less efficient than periodic conditions when applied to the canonical method, because the simulation edges must be reset after each diffusion step. When applied to the performance enhancement, however, fixed boundary conditions are more efficient because the modulus operator is no longer needed.
CHAPTER 5

Identifying Steady-State Areas

This chapter outlines a method, developed by [1], for identifying stable or steady-state tissue areas in the simulation from [11] (described in Chapter 3). Identification of steady-state areas is necessary to implement the performance enhancement described in Chapter 6, which updates steady-state areas less often than unstable areas. The enhancement is based on the hypothesis that updates to rapidly-changing areas are more crucial to maintaining simulation realism than updates to stable areas. A defense of this hypothesis is presented in Section 1.2. This chapter describes the implementation of a procedure to identify steady-state areas in the simulated tissue, so that updates to those areas may be repressed by the performance enhancement.

1. Adding Steady-State Identification Functionality

Tissue areas which meet the following criteria are defined as steady-state:

(1) Oxygen concentration change must be below a specific threshold $\varepsilon_O$. Similarly, glucose and proton concentration changes must be below thresholds $\varepsilon_G$ and $\varepsilon_H$.

(2) Population change of all cell types must be zero.
In order to decide which tissue areas are steady-state, local changes in cell populations and nutrient concentrations must be tracked for each grid element. When the entire grid is updated to advance the simulation by one timestep, all previous local concentrations are remembered; the previous values are then subtracted from the newly-calculated values to obtain six matrices $E_n$, $E_t$, $E_r$, $E_O$, $E_G$, and $E_H$ containing the change in nutrient concentration and cell population in each grid element over the last timestep.

Grid elements are labeled as steady-state or unstable in the following manner. First, the matrices $E_n$, $E_t$, and $E_r$ are checked to ensure that cell population change in the grid element of interest is zero. If cell population change is greater than zero, the element is labeled as unstable. Otherwise, all elements of the three matrices are then compared to their corresponding threshold values. Let $E_M$ be the concentration change matrix for nutrient $M$. If the absolute value of element $(x, y)$ in $E_M$ is below $\varepsilon_M$, then grid element $(x, y)$ is considered steady-state with respect to nutrient $M$. Element $(x, y)$ in an additional matrix $S_M$ is then set to 0 in order to mark tissue location $(x, y)$ as steady with respect to $M$. In this way, three matrices $S_O$, $S_G$, and $S_H$ are obtained, with values of zero indicating steady areas with respect to a nutrient, and values of one indicating unstable areas. These three matrices label grid elements as steady-state or unstable with respect to oxygen, glucose, and protons.
2. LOCATIONS OF STEADY-STATE AREAS

Figure 1. Locations of steady-state areas (shown in red) with respect to oxygen in a 252-by-252 simulation using different settings of the oxygen steady-state threshold. \( \varepsilon_O \) is equal to \( 9 \times 10^{-7} \) on the left and \( 9 \times 10^{-6} \) on the right. The higher setting causes a larger tissue area to be declared steady.

2. Locations of Steady-State Areas

Depending on the steady-state threshold settings (\( \varepsilon_O, \varepsilon_G, \varepsilon_H \)), continuous regions of the simulated tissue are labeled as steady state. In Figures 1 and 2, the \( S_O \) matrix is plotted at various simulation timesteps, illustrating the locations of steady and unstable areas with different settings for \( \varepsilon_O \).

As shown in Figure 2, with certain values for \( \varepsilon_O \), the outer tissue areas are declared steady, while the areas closer to the tumor are declared unstable. This pattern occurs while the tumor is growing rapidly; it suggests that tumor cells are the agents of simulation instability.

As tumor growth continues, the boundary between stable and unstable tissue locations expands outward, until the entire tissue is eventually declared unstable. This phase of tumor growth illustrates that the tumor creates a concentration
2. LOCATIONS OF STEADY-STATE AREAS

Figure 2. Locations of steady-state areas (shown in red) with respect to oxygen in a 252-by-252 simulation at 42 and 290 timesteps. Steady-state thresholds are as follows: \( \varepsilon_O = 9 \times 10^{-6} \) millimolar, \( \varepsilon_G = 4 \times 10^{-4} \) millimolar, and \( \varepsilon_H = 5 \times 10^{-9} \) millimolar. During rapid tumor growth, unstable areas are located near the tumor; when central tumor cell growth halts, the tumor interior is declared steady-state.

gradient of oxygen and glucose, which flow inward from the more nutrient-rich outer areas. Nutrient flow rate increases as the concentration gradient steepens with greater proximity to the tumor. At first, nutrient flow in areas near the tissue’s edge is slow, but as more nutrients in the central region are consumed, the magnitudes of centrally-directed concentration gradients increase throughout the tissue, and nutrient flow rate therefore increases in areas distant from the tumor. This causes the central, unstable tissue area to expand outward.

The tumor eventually reaches a maximum sustainable radius, and expansion halts. Due to the low nutrient levels in the tumor’s interior, tumor cells are unable to reproduce, and cell populations stabilize. Because the nearest reproducing
tumor cells are near the tissue’s edge, concentrations in the tissue center stabilize, and the central region is declared steady. As tumor cells reach stable populations and stop growing, tissue-wide nutrient concentrations begin to stabilize, and the stable central area expands outward until the end of the simulation.

3. Discussion of Steady-State Locations

A notable result is that the steady and unstable areas are mainly continuous and separate, suggesting that the performance enhancement developed by [1] may improve simulation performance by suppressing updates to these areas. Large areas of steady-state tissue border similar regions of unstable tissue; stable and unstable areas are not randomly interspersed throughout the tissue. This suggests that unstable nutrient concentrations propagate outward from the original locations of nutrient concentration change. The location of unstable areas near tumor cells illustrates that tumor cells are the originators of nutrient concentration change, with their rapid population growth and nutrient consumption.

In general, the results of steady-state identification show that the simulated tissue can be classified into large steady and unstable areas, suggesting that performance could be increased by suppressing updates to unstable areas. The implementation of this enhancement would require frequent access to the matrices $S_O$, $S_G$, and $S_H$ to apply update suppression to the appropriate locations, and to mark locations for future suppression of updates. To speed access to the $S$ matrices, a hierarchical data structure, described in the next chapter, was designed.
CHAPTER 6

Enhancing Simulation Performance by Suppressing Updates to Steady-State Areas

This chapter describes a performance enhancement, developed by [1], to the cellular automaton-based simulation from [11]. The enhancement updates steady-state areas less often than stable areas, possibly improving efficiency by reducing the total number of grid element updates.

The implementation of the performance enhancement is divided into two tasks. First, steady-state tissue locations (defined using the criteria from Chapter 5) are marked using a hierarchical data structure (Section 6.1). Secondly, the tissue is updated in small blocks rather than globally; unstable blocks are updated every timestep as in the original implementation, but updates to steady-state blocks are suppressed. Every $k$ timesteps, the entire tissue is updated, including steady-state areas.

1. Hierarchical Representation of Steady-State Areas

A hierarchical data structure is used to mark tissue areas labeled as steady-state using the method developed by [1] (Chapter 5). Three matrices are used: $S_O$, $S_G$, and $S_H$. If the simulated tissue has side length $N$, then $S_O$, $S_G$, and $S_H$ are $N$-by-$N$ matrices. Entries in an $S_x$ matrix (with $x \in \{O, G, H\}$) correspond to...
specific tissue locations: element \((x, y)\) in \(S_x\) corresponds to element \((x, y)\) in the simulated tissue. Entries of 0 in \(S_x\) indicate that the corresponding tissue location is steady-state with respect to nutrient \(x\); that is, the concentration change of \(x\) is below threshold \(\varepsilon_x\), where \(\varepsilon_x\) is a steady-state threshold from Chapter 5. Entries of 1 in \(S_x\) label unstable locations with respect to nutrient \(x\). The arrays \(S_x\) must be accessed many times throughout the simulation, to update them with the locations of new steady-state and unstable areas, and to determine the location of current steady-state areas for suppression of updates.

To reduce the total number of \(S\) matrix accesses required, a hierarchical data structure is proposed [1]. Two additional boolean matrices are declared for each nutrient \(x\): \(S_{3x}\), with side length \(\frac{N}{3}\), and \(S_{9x}\), with side length \(\frac{N}{9}\). The matrices have a hierarchical relationship: \(S_{3x}\) is a low-resolution view of \(S_x\), and \(S_{9x}\) is a low-resolution view of \(S_{3x}\). That is, elements of \(S_{3x}\) represent 3-by-3 neighborhoods in \(S_x\); elements of \(S_{9x}\) represent 3-by-3 neighborhoods in \(S_{3x}\) and 9-by-9 neighborhoods in \(S_x\). Specifically, element \((r, c)\) of \(S_{3x}\) represents the 9-element 3-by-3 neighborhood \((3r..3r + 2, 3c..3c + 2)\). Element \((r, c)\) of \(S_{9x}\) represents the 3-by-3 neighborhood \((3r..3r + 2, 3c..3c + 2)\) of \(S_{3x}\) and the 9-by-9 neighborhood \((9r..9r + 8, 9c..9c + 8)\) of \(S_x\).

Entries in \(S_{3x}\) indicate whether or not all grid elements in the corresponding neighborhood of \(S_x\) are steady-state. Entries of 0 in \(S_{3x}\) indicate that the corresponding neighborhood in \(S_x\) is entirely steady-state; entries of 1 indicate that at least one grid element in the corresponding neighborhood is unstable. Similarly,
entries in $S_{9x}$ indicate whether or not the corresponding neighborhoods in $S_{3x}$ are steady-state.

Elements of matrix $S_{3x}$ are set as follows:

$$S_{3x}(r,c) = \begin{cases} 
0 & \text{if } S_{x}(a,b) = 0 \text{ for each } (a,b) \in \text{nbhd3}(r,c), \\
1 & \text{otherwise}
\end{cases}$$ \hspace{1cm} (3)

In Equation 3, $\text{nbhd3}(r,c)$ represents the 3-by-3 neighborhood of $S_O$ corresponding to element $(r,c)$ of $S_{3x}$.

Elements of $S_{9x}$ are set in a similar manner:

$$S_{9x}(r,c) = \begin{cases} 
0 & \text{if } S_{3x}(a,b) = 0 \text{ for each } (a,b) \in \text{nbhd3}(r,c) \\
1 & \text{otherwise}
\end{cases}$$ \hspace{1cm} (4)

where $\text{nbhd3}(r,c)$ represents the 3-by-3 neighborhood of $S_{3x}$ corresponding to element $(r,c)$ of $S_{9x}$. Because the element-to-neighborhood correspondence between an $N$-by-$N$ matrix and an $N/3$-by-$N/3$ matrix is equivalent to the correspondence between an $N/3$-by-$N/3$ matrix and a $N/9$-by-$N/9$ matrix, the $\text{nbhd3}$ functions in Equations 3 and 4 are equivalent. Figure 1 shows an example setting of $S_O$, $S_{3O}$, and $S_{9O}$.

Elements of $S_{3x}$ are zero if and only if all elements in their corresponding neighborhood of $S_x$ are zero, and elements of $S_{9x}$ are zero if and only if all elements in their corresponding neighborhood of $S_{3x}$ are zero. Equations 3 and 4 also imply that elements of $S_{9x}$ are zero if and only if all elements in their corresponding 9-by-9 neighborhood of $S_x$ are zero.
Figure 1. An example setting of $S_O$, $S_{3O}$, and $S_{9O}$ for a 9-by-9 tissue.

2. Implementation of Partial Grid Updates

This section describes a performance enhancement to the simulation from [11], in which steady-state tissue areas, identified using the criterion from Chapter 5 and tracked using the data structure from Section 6.1, are updated less often than unstable areas. Implementation of the enhancement required modification of the simulation’s original tissue-wide update routine to operate on small blocks of tissue, rather than on the entire tissue. This allows the tissue to be selectively updated each timestep; steady-state areas can be omitted, improving simulation efficiency.

The simulation from [11], which is written in MATLAB, stores the state of the $N$-by-$N$ simulated tissue in six $N$-by-$N$ matrices: one for each cell type and nutrient. The six matrices are updated every timestep according to the simulation rules presented in Chapter 3. Nutrient diffusion operations are applied to...
the entirety of the three nutrient concentration matrices, and each cell population matrix is completely updated using the cell growth rules. These updates are done using MATLAB’s built-in matrix operations, which update an entire matrix in a single programming step, so the partial matrix updates required by the performance enhancement were not available using this method.

Modification of the original code was needed to enable partial matrix updates; two approaches were tested. First, the simulation code was rewritten to update the six matrices one grid element at a time, allowing steady-state elements to be skipped. The overhead required to implement this method resulted in a significant speed decrease—suppression of updates to large areas of tissue were unlikely to compensate for the slow program speed. This indicated that MATLAB’s built-in matrix operations are more efficient than loop-based operations to single elements in MATLAB; use of the built-in operations should therefore be maximized.

Secondly, the code was rewritten to update small blocks of matrix elements rather than individual elements. This method uses MATLAB’s built-in matrix operations more heavily than the first method, improving computational efficiency. The method works as follows: First, the hierarchical data structure is set based on the locations of steady-state areas, as described in the previous section. Then, $S9_x$ is checked to find 9-by-9 neighborhoods that require updates to nutrient $x$. In the previous step, such neighborhoods were marked with 1 in $S9_x$, because they contain unstable elements. Then, for each 9-by-9 block $B$ that requires an update, $S3_x$ is checked to determine which of $B$’s constituent 3-by-3 blocks require
2. IMPLEMENTATION OF PARTIAL GRID UPDATES

Figure 2. Two examples of the $S_O$ matrix for a 9-by-9 tissue. In (A), all constituent 3-by-3 neighborhoods contain unstable elements and therefore require updates, so the entire 9-by-9 tissue (outlined in red) is updated using MATLAB’s built-in matrix operations. In (B), the constituent 3-by-3 neighborhoods outlined in red require updates; the other neighborhoods do not. The tissue blocks outlined in red are therefore updated using MATLAB’s built-in operations.

updates. If all of the constituent 3-by-3 blocks require updates (i.e., the relevant elements in $S_{3_x}$ are all 1), then the entire 9-by-9 neighborhood is updated using MATLAB’s built-in matrix operations. If at least one constituent 3-by-3 block does not require an update, the remaining 3-by-3 blocks are individually updated using the built-in matrix operations (Figure 2).

This method allows the tissue to be updated in 9-by-9 blocks when possible, reducing the amount of looping required by maximizing the amount of tissue updated per iteration. When the presence of stable tissue areas prevents updates over an entire 9-by-9 block, updates are done in 3-by-3 blocks instead. If a 3-by-3
block contains steady and unstable elements, the entire block is updated; thus, updates are never applied to single grid elements. By updating tissue in small blocks rather than globally, the method applies updates only to unstable areas, improving performance if the percentage of steady-state areas is sufficiently high.

3. Discussion of Performance Enhancement

3.1. Stability of Nutrient Concentration Change. The performance enhancement is unlikely to provide accurate results if nutrient concentration change is unstable (i.e., if concentrations can remain steady for several timesteps, then suddenly change rapidly). If this occurred in a location where update suppression was occurring, the difference in results between the two simulation methods would be high. However, simulations are not likely to have this property, because tumor cells are the agents of simulation instability.

Actively-dividing tumor cells prevent the simulation from entering a steady state. Cancer-free simulations reach a steady state in less time than simulations incorporating tumor cells: Healthy cells in a tumor-free simulation quickly reach a sustainable population, remaining at that population indefinitely as time passes and nutrient concentration change approaches zero. Furthermore, dividing tumor cells in a given location increase nutrient concentration change in the surrounding region. Because local cell populations are a function of local nutrient concentrations, the presence of tumor cells also affects cell populations in the
region. Therefore, tumor cells can be considered the agents of simulation instability: Their presence delays the onset of steady-state conditions and causes local changes in nutrient concentration and cell population.

Because tumor cells cannot spontaneously appear in arbitrary locations, tissue instability induced by tumor cells is unlikely to drastically decrease the performance enhancement’s accuracy. New tumor cells may only enter the simulation as a consequence of cell division by preexisting tumor cells; tumor cells may not enter the simulation by metastasizing from other tissues. As a result, new tumor cells can appear at most one grid element away from preexisting tumor cells. Because tumor cells are the cause of nutrient concentration instability, and tumor cells cannot spontaneously appear in a previously stable location, nutrient concentrations are unlikely to suddenly become unstable. Instead, locations with stable nutrient concentrations will slowly display more instability as the edge of the growing tumor approaches. Sudden, rapid change in nutrient concentrations is therefore unlikely to appear in grid elements that are not being updated; as long as the time $k$ between updates to steady-state areas is small enough, the gradual onset of instability will cause previously steady-state areas to be declared unstable as tumor cells approach, ending suppression of updates. The performance enhancement’s accuracy is therefore unlikely to be decreased by sudden changes in locations that are not being updated.

3.2. Efficiency of Hierarchical Data Structure. Use of the hierarchical data structure to update tissue areas requires fewer matrix accesses than if $S_O$,
3. DISCUSSION OF PERFORMANCE ENHANCEMENT

$S_G$, and $S_H$ were used alone, without the hierarchy of $S3_x$ and $S9_x$ matrices. For example, if an element of $S9_x$ is zero, then the corresponding 9-by-9 tissue block is entirely steady and will not be updated, so no further matrix accesses are required for that tissue block. In contrast, if the hierarchical data structure were not used, 81 array accesses—one for each grid element in the 9-by-9 block—would be needed in order to confirm that none of the neighborhood’s constituent elements required updates.

The hierarchical data structure reduces matrix accesses when assigning updates to tissue areas, but the number of required accesses is increased when updating the data structure with the locations of new steady areas. Every $k$ timesteps the entire tissue is updated and relabeled as steady or unstable. With the hierarchical data structure, $S_x$, $S3_x$, and $S9_x$ must all be completely updated, requiring $N^2 + \frac{N^2}{9} + \frac{N^2}{81}$ matrix accesses per nutrient, where $N$ is the tissue side length; without the data structure, only $N^2$ accesses per nutrient are required. Overall, the hierarchical data structure saves computational time if block-based partial tissue updates occur often enough with respect to tissue-wide relabeling of steady and unstable areas.

3.3. Efficiency of Performance Enhancement. Experiments show that the performance enhancement can be more efficient (as measured by program

Footnote 1: According to [24], a reasonable value for $N$ is 500; this corresponds to a simulated tissue area of approximately 4.5 cm$^2$. Due to efficiency concerns, smaller values for $N$ were used for the experiments reported in this paper.
running time) than the canonical simulation method if the steady-state thresholds $\varepsilon_O$, $\varepsilon_G$, and $\varepsilon_H$ are sufficiently high. For example, on a 1.86 GHz processor with 2 GB RAM, the original simulation from [11] requires 127 seconds to simulate tumor growth in a 252-by-252 tissue for 50 timesteps, but the performance enhancement described by [1] requires 47 seconds with specific threshold settings. If the thresholds are too low, the performance enhancement runs less efficiently than the original method, because updating large portions of the grid in 9-by-9 and 3-by-3 blocks is less efficient than updating the entire grid in one step. For instance, when run on the same tissue used for the above experiment, the performance enhancement described by [1] requires 296 seconds to simulate 50 timesteps with threshold settings lower than those in the previous experiment.

As demonstrated above, the performance enhancement described by [1] can improve computational efficiency in some situations but not in others. Efficiency concerns therefore affect the decision of whether to use the performance enhancement designed by [1] or the original method from [11]. Because the performance enhancement produces different results from the original method, accuracy should also be taken into account when choosing a simulation method. The next chapter provides criteria for judging the accuracy of the performance enhancement.

---

$^2\varepsilon_O = 1.1 \times 10^{-4}$ millimolar; $\varepsilon_G = 1.2 \times 10^{-4}$ millimolar; $\varepsilon_H = 7 \times 10^{-10}$ millimolar.

Grid-wide updates are forced every 10 timesteps.

$^3\varepsilon_O = 9 \times 10^{-10}; \varepsilon_G = 4 \times 10^{-8}; \varepsilon_H = 5 \times 10^{-13}$. Grid-wide updates are forced every 10 timesteps.
Chapter 7

Criteria for Equivalence Between Simulations

The performance enhancement outlined in Chapter 6 produces different results from the original simulation method from Chapter 3; criteria are therefore needed to decide whether or not the enhancement can be considered equivalent to the original method. In general, raising the steady-state threshold values $\varepsilon_O$, $\varepsilon_G$, and $\varepsilon_H$ increases computational efficiency; lowering the thresholds reduces efficiency and produces results closer to those of the canonical simulation. The performance enhancement should be used to increase efficiency as much as possible without producing results that are meaningfully different from those of the original, canonical simulation method. To make this judgment, criteria are necessary by which modified simulation methods (such as the performance enhancement from Chapter 6) can be considered equivalent to a canonical method (e.g., the original simulation), described in Chapter 3.

Three such equivalence criteria are discussed in this chapter. The criteria take the following parameters as input: a modified simulation method $M$, a canonical method $C$, a start state $S$, and a timespan $t_f$. Given this information, a function can determine whether the results of running $M$ on $S$ for $t_f$ timesteps are equivalent to those produced by running $C$ on $S$ for $t_f$ timesteps, where equivalent is defined by the criteria discussed below.
1. Criterion Based on Cell Populations

The first equivalence criterion declares a modified simulation to be equivalent to a canonical method if and only if the two simulations are everywhere identical in cell population, i.e., the following statement is true:

Criterion 1. The tissue-wide cell populations produced by running $M$ on $S$ are equal to those produced by running $C$ on $S$ at all timesteps $t$, with $t \leq t_f$.

1.1. Theoretical Justification. The performance enhancement from Chapter 6 produces different nutrient concentrations from the canonical method but does not necessarily cause a difference in cell populations. Nutrient concentrations and cell populations are the two main dynamic components of the simulation; nutrient concentrations vary continuously, but cell populations vary discretely. The performance enhancement necessarily produces differences in nutrient concentrations, but small nutrient concentration differences might not be large enough to change cell populations by even a single cell, so it is possible that no corresponding difference in cell populations would be induced.

Implicit in this criterion is the assumption that nutrient concentrations themselves are relevant only in their effects on the simulation result of interest, cell populations. This assumption is reasonable if, for example, the simulation is used to predict the future growth rate and pattern of a tumor. In this application, cell populations and their spatial distributions are relevant, but nutrient concentrations are not. The assumption is not reasonable when nutrient concentrations are
1. CRITERION BASED ON CELL POPULATIONS

central to the application of the simulation. For example, Criterion 1 should not be used if the simulation is used to study nutrient deprivation to tumors.

1.2. Implementation. A function was implemented to take $M$, $C$, $S$, and $t_f$ as input and decide whether $M$ is equivalent to $C$ using Criterion 1. The function runs two simulations: one using method $M$ (the performance enhancement), and one using method $C$ (the canonical method). Both simulations are run for $t_f$ timesteps, using $S$ as starting conditions. The two simulations’ cell population histories are then compared; the simulations are declared equivalent if and only if their cell populations are equal in all locations at all timesteps.

1.3. Equivalence Between Performance Enhancement and Canonical Method. From the start state developed in Section 4.2, simulations incorporating the performance enhancement from Chapter 6 were run with different steady-state thresholds ($\varepsilon_O$, $\varepsilon_G$, $\varepsilon_H$) and tested for equivalence with the canonical simulation run on the same starting conditions. The results showed that, even with low threshold values that caused little suppression of updates, performance-enhanced simulations have different cell population histories and are therefore not equivalent to the canonical method by Criterion 1. For example, Figure 1 illustrates cell population error in a simulation of a 54-by-54 tissue with low threshold settings. Steady-state threshold settings are so low that updates are only suppressed in a single 3-by-3 neighborhood at the tissue’s edge (shown in Figure 2) for 5 of the 700 total timesteps. Update suppression begins at approximately timestep 20, introducing a difference in nutrient concentrations of less than $1 \times 10^{-9}$ millimolar.
Cell populations remain identical to the canonical simulation until approximately timestep 150, when a small, temporary deviation in tumor cells occurs. Many more timesteps pass with identical cell populations, until approximately timestep 350, when a deviation in tumor and necrotic cell populations occurs and increases, reaching a maximum difference of 120 tumor cells by the end of the simulation. Figure 3 shows the grid elements where tumor cell populations vary between the normal and canonical simulations.

1.4. Discussion. The above results demonstrate that small deviations in nutrient concentration can cause increasing differences in cell population, indicating the presence of simulation instability that prevents performance-enhanced simulations from being declared equivalent to the canonical method by Criterion 1. Performance-enhanced simulations may still provide valuable information about tumor growth. For example, the 120-cell population difference in the performance-enhanced simulation mentioned above is an average deviation of only 0.04 tumor cells per grid element, an intuitively insignificant difference; nonetheless, the criterion discussed in this section does not consider this simulation equivalent to the canonical method. Less strict criteria, such as those described in Sections 7.2 and 7.3, may therefore be more useful.

2. Criterion Based on Measurements of Tissue Start State

The second equivalence criterion is based on the inherent inaccuracy of biological measurements. As with Criterion 1, this criterion is defined with respect to
a tissue-wide start state \( S \), which is considered to be a measurement of an actual biological tissue sample. Because biological measurements have an associated error range, the measurements used to obtain the start state can be inaccurate. For example, if measurements of cell populations in biological tissue can be up to \( e\% \)
Figure 2. Update suppression in the performance-enhanced simulation with epsilon settings reported in Figure 1 occurs only in the area indicated in this figure. The 54-by-54 tissue is shown in gray; the black box in the middle right represents a single 3-by-3 neighborhood which was not updated for five timesteps, beginning at approximately timestep 20.

Inaccurate, and such measurements determine a tissue’s cell population to be $p$ at location $q$, then the actual cell population at $q$ could be any value within the range $E = [p - \frac{\epsilon_{100} p}{100}, x + \frac{\epsilon_{100} p}{100}]$. A start state $S_q$ for location $q$ with cell population $p$ would be defined in order to simulate this tissue; however, any start state for $q$ with cell population in the range $E$ could reasonably be used, because the actual tissue’s cell population at $q$ is some value within $E$. Thus, given a tissue-wide start state $S$ and an error range $e$, a set $R$ of reasonable tissue-wide start states can be defined.
A set of reasonable tissue-wide start states can be created by varying cell populations in the original tissue-wide start state $S$ by up to $e\%$. When run on these states for $t_f$ timesteps, the canonical simulation produces a set of reasonable tissue-wide result states at each timestep $t$ up to $t_f$. The modified simulation is considered equivalent to the canonical method if, when run on the original tissue-wide start state $S$, its results are within the set of reasonable tissue-wide result states at each timestep. The relevant sense of “within” is explained in Sections 7.2.1 and 7.2.2.
2. CRITERION BASED ON MEASUREMENTS OF TISSUE START STATE

2.1. Theoretical Justification. The criterion considers a modified method 
$M$ to be equivalent to a canonical method $C$ with respect to a timespan $t_f$ and
a start state $S$ if and only if the modified method’s cell population results are
within the set of reasonable tissue-wide result states of $C$, considering that there
are a number of reasonable start states that differ slightly from $S$.

Tissue starting conditions are obtained from some source (presumably, cell
population measurements of a biological tissue sample) and input into the simu-
lation, which returns the simulated tissue state up to time $t_f$. At a given cellular
automaton grid element $q$, let $S_q = (n_q, u_q, r_q)$ be the starting healthy, tumor, and
necrotic cell populations, obtained from a biological measurement with error range
e\%, at grid element $q$. State $S_q$ can be visualized as a point in three-dimensional
space with dimensions for healthy, tumor, and necrotic cell population. Due
to measurement inaccuracy, the actual biological tissue state could be anywhere
within the 3-box $R_q$, defined as below:

$$R_q = \left\{ (n'_q, u'_q, r'_q) \mid \frac{|n'_q - n_q|}{n_q} \leq \frac{e}{100}, \frac{|u'_q - u_q|}{u_q} \leq \frac{e}{100}, \frac{|r'_q - r_q|}{r_q} \leq \frac{e}{100} \right\}.$$  \hspace{1cm} (5)

For purposes of equivalence criteria, $R_q$ is the set of reasonable start states at
tissue location $q$ based on the measurement result $S_q$ and the error range $e$.

$S_q$ and $R_q$ are defined for each grid element $q$. Tuple $GR$ (for global $R$) is
defined as follows:

$$GR = (R_q \text{ for all grid elements } q)$$  \hspace{1cm} (6)

Thus, $GR$ contains the set of reasonable start states for each grid element.
2. CRITERION BASED ON MEASUREMENTS OF TISSUE START STATE

Let \( GGP \) denote the set of all reasonable tissue-wide start states, i.e., tissue states in which each grid element is an individual reasonable start state. \( GGP \) is a set of tuples \( GP \) with the following form:

\[
GP = (P_1, P_2, \ldots, P_{N^2}) \text{ such that } P_q = (n'_q, u'_q, r'_q) \in R_q
\]  

(7)

where \( N \) is the tissue side length, and grid elements \( q \) are numbered from 1 to \( N^2 \). Thus, \( GP \) is a single, tissue-wide reasonable start state consisting of cell population triples \( P_q = (n'_q, u'_q, r'_q) \) for all grid elements \( q \).

The canonical simulation method \( C \) is a function that maps tissue-wide start states to tissue-wide result states up to a particular time \( t_f \). Thus, \( C \) can be run on a reasonable tissue-wide start state \( GP \) in Equation 7 for \( t \) timesteps to produce \( GD_t \), a reasonable tissue-wide result state at time \( t \):

\[
C(GP, t) = GD_t = (D_{t,1}, D_{t,2}, \ldots, D_{t,N^2})
\]  

(8)

where \( D_{t,q} = (n'', u'', r'') \) is the state of grid element \( q \) at time \( t \), \( N \) is the tissue side length, and grid elements \( q \) are numbered from 1 to \( N^2 \). If \( C \) is run sequentially on all tissue-wide reasonable start states in the set \( GGP \), the set \( GGD_t \) of all tissue-wide reasonable result states at time \( t \) is generated.

\[
GGD_t = \{C(GP, t) \text{ such that } GP \in GGP\}
\]  

(9)

The modified simulation method \( M \) is also a function that maps a tissue-wide start state \( S \) to a tissue-wide result state at time \( t \):

\[
M(S, t) = (B_{t,1}, B_{t,2}, \ldots, B_{t,N^2})
\]  

(10)
Where \( t < t_f \) and \( B_{t,q} = (n'', u'', r'') \) is the state of grid element \( q \) at time \( t \), \( N \) is the tissue side length, and grid elements \( q \) are numbered from 1 to \( N^2 \).

Finally, the equivalence criterion considers a modified simulation \( M \) to be equivalent to a canonical simulation \( C \) with respect to start state \( S \) over timespan \( t_f \) if and only if, at all timesteps up to \( t_f \), \( M \)'s tissue state is in the set of reasonable tissue-wide result states of \( C \). That is, \( M \) is equivalent to \( C \) if and only if Criterion 2 holds:

**Criterion 2.** \( \forall 1 \leq t \leq t_f \) \( M(S, t) \in \text{GGD}_t \)

To summarize, a function deciding simulation equivalence based on the above criterion takes four parameters: a canonical simulation method \( C \), a modified method \( M \), a timespan \( t_f \), and a starting condition \( S \). First, the function generates the set \( GGP \) of all reasonable tissue-wide start states, based on the original start state \( S \) and an error range \( e \) of biological measurements. Using the canonical method, it maps \( GGP \) to sets of reasonable tissue-wide result states at all timesteps up to \( t_f \). It then runs the modified method on \( S \) to obtain a tissue-wide result state \( M(S, t) \) at each timestep up to \( t_f \). The function returns true (indicating that the modified simulation is equivalent to the canonical method) if and only if, at each timestep up to \( t_f \), the modified simulation’s tissue-wide state is in the set of reasonable tissue-wide result states of the canonical simulation.
2.2. Implementation. The theoretical justification above uses individual grid elements as the basic spatial unit for judging the equivalence of two simulations. In fact, the tissue can be divided into $w$-by-$w$ blocks (for a specific $w$), which can then be treated as individual grid elements $q$ by totaling the cell population in each $w$-by-$w$ block. This approach is taken (with $w$ equal to 9) in the implementation described below, for reasons that are explained in Section 7.2.4.1. Therefore, in the following discussion, the variable $q$ represents $w$-by-$w$ tissue blocks over which cell populations are totaled, rather than single grid elements.

Implementation of a procedure for testing simulation equivalence required a modification to the theoretical criterion, because computation of the tuple $(GGD_1, GGD_2, \ldots, GGD_{t_f})$, the sets of all reasonable tissue-wide result states at times 1 to $t_f$, is infeasible due to the high number of such sets. As a result, a small number of reasonable tissue-wide result states is computed for each timestep, and geometric techniques are used to estimate the remainder of the set.

Given a start state $S$ and error range $e$ (10\% in the current implementation [5]), a random subset of $GGP$, the set of reasonable tissue-wide start states, is generated. To accomplish this, the tissue is divided into adjacent 9-by-9 blocks; total cell population in each block is randomly chosen to increase or decrease by up to $e\%$. If, for example, a $p$-cell population increase is chosen, $p$ cells are added, distributed over random elements in the 9-by-9 block while ensuring that no grid element exceeds the maximum allowed population of 100 cells. If a $p$-cell population decrease is chosen, $p$ cells are removed, distributed over random grid elements.
in the neighborhood. The resulting tissue state is a reasonable tissue-wide start state with the form of $GP$, because the total cell population in each 9-by-9 block has been varied by no more than $\epsilon\%$ of its original value. The cell population randomization of $S$ is then repeated $z$ times ($z$ is 25 in the current implementation); different stochastic decisions are made in each iteration, producing a set \{\(GP_1, GP_2, \ldots, GP_z\)\} of $z$ reasonable tissue-wide start states with the form of $GP$ (Equation 7); The canonical method is then run on each of the reasonable tissue-wide start states $GP_i$ for $t_f$ timesteps ($t_f$ is currently 700) to produce $z$ reasonable tissue-wide result states ($GD_{t,1}, GD_{t,2}, \ldots, GD_{t,z}$) for each $t$ from 1 to $t_f$.

A three-dimensional plot with axes for healthy, tumor, and necrotic cell populations is created for each 9-by-9 tissue block $q$ at each timestep $t$. A point is plotted for each of the $z$ reasonable tissue-wide result states $GD_{t,i}$ for $i$ from 1 to $z$; points represent the total cell population in block $q$ at time $t$ of a particular $GP_{t,i}$. These plots form collections of points in three-dimensional space. As explained above, many of the reasonable tissue-wide result states in $GGD_t$ were not calculated. The points for these states are assumed to be located near the plotted points for the $z$ reasonable tissue-wide result states that were calculated; the convex hull of the $z$ points is taken, and the resulting region $PH_{q,t}$ is considered an estimate of the set of all reasonable result states at timestep $t$ and tissue block $q$. 

The modified simulation method \( M \) is then run on the original tissue-wide start state \( S \) for \( t_f \) timesteps. For each adjacent 9-by-9 tissue block \( q \) at each timestep \( t \), the block’s total cell population is plotted as a point \( U_{q,t} \) in three-dimensional space, with dimensions for healthy, tumor, and necrotic cell population. If \( U_{q,t} \) is within the region \( PH_{q,t} \), then the modified simulation is considered equivalent to the canonical method in tissue block \( q \) at time \( t \). The entire modified simulation is considered equivalent to the canonical method if and only if all neighborhoods \( q \) are equivalent at all timesteps \( t \) in \( q..t_f \).

2.3. Equivalence Between Performance Enhancement and Canonical Method. Equivalence tests using Criterion 2 are accomplished by constructing three-dimensional regions, based on a set of reasonable tissue-wide result states, for each 9-by-9 tissue block \( q \) at each timestep \( t \). Regions \( PH_{q,t} \) constructed for various tissue blocks \( q \) and timesteps \( t \) have been observed to take a variety of forms, some of which are shown in Figures 4–5.

Furthermore, the region \( PH_{q,t} \) for a given 9-by-9 tissue block \( q \) can vary substantially with timestep \( t \). Figure 4 illustrates the region for the 9-by-9 tissue block in the center of full tissue at different timesteps. The rate of geometric change of equivalence regions is not always constant: Some regions change rapidly over certain timespans and slowly over others. For example, the central equivalence region shown in Figure 4 begins with high healthy population and low tumor population. The region moves in the direction of positive tumor growth, until approximately timestep 60, when massive healthy cell death causes the region to
Figure 4. Equivalence regions for the central 9-by-9 tissue block in a 252-by-252 tissue. 25 reasonable result states, and the canonical simulation’s result state, were used to create these regions. Timesteps shown are 50, 146, 365, and 400 (see lower left corner of images). When the canonical result state is at the edge of the equivalence region, it is shown in red.

move rapidly to zero healthy cells. The region then coalesces to a single point with 8,000 tumor cells and no other cell types, remaining in this position for approximately 150 timesteps. At timestep 300, tumor cell death begins, causing the region to expand into a two-dimensional polygon and move rapidly in the direction of positive necrotic cell growth and tumor cell death. Finally, necrotic cell decay begins, and the region moves toward lower necrotic population values. By
Figure 5. Equivalence region for the upper left 9-by-9 tissue block in a 252-by-252 tissue at timestep 100. As in Figure 4, 25 reasonable result states and the canonical result state were used to create the region. The canonical data point is shown in red.

timestep 450, the tissue center has entered a steady state, and the equivalence region remains constant for the remainder of the simulation.

These periods of rapid and slow change are induced by the simulation states from which the convex hull was constructed. At most timesteps, tissue state at a given location is stable. However, sudden events such as invasion by tumor cells, or massive tumor cell death to form a necrotic core, produce rapid changes in tissue state.

As shown in Figures 4–5, the equivalence regions can be more inclusive than those of the criterion from Section 7.1, which can be described as single points corresponding to the state of the canonical simulation, rather than regions based
on a number of reasonable result states. Nevertheless, in all tested cases Criterion 2 considers performance-enhanced simulations with very low steady-state threshold settings \((\varepsilon_O, \varepsilon_G, \varepsilon_H)\) to be non-equivalent to the canonical method. For example, a performance-enhanced simulation with steady-state threshold settings \((\varepsilon_O = 9 \times 10^{-6} \text{ millimolar}, \varepsilon_G = 4 \times 10^{-4} \text{ millimolar}, \varepsilon_H = 5 \times 10^{-9} \text{ millimolar})\) and tissue-wide updates forced every 5 timesteps runs becomes non-equivalent at timestep 57 in the central tissue block. The non-equivalent area spreads outward from the center, forming concentric rings of equivalent and non-equivalent 9-by-9 tissue blocks (Figure 6).

When threshold settings are lowered below those used in Figure 6, simulations run using the performance enhancement are still declared non-equivalent. Figure 7 illustrates the problem. The modified simulation’s tumor cell population in one neighborhood increases to one cell greater than the canonical simulation’s population. This moves the modified simulation’s data point one unit right of the canonical data point. Because the canonical data point is a vertex of the equivalence region, this displacement is sufficient to move the modified simulation’s point outside the region, causing the simulation to be declared non-equivalent to the canonical method. This may indicate that Criterion 2 is too strict, as a single-cell population discrepancy does not intuitively seem to be a meaningful difference between two simulations.
2. CRITERION BASED ON MEASUREMENTS OF TISSUE START STATE

2.4. Discussion.

2.4.1. Use of Tissue Blocks for Equivalence Test. The theoretical definition of the criterion (Section 7.2.1) uses single grid elements rather than 9-by-9 tissue blocks as the basis for determining equivalence. This can produce undesirable results if the modified simulation’s cell populations are identical to the canonical populations except for a small spatial translation. If single grid elements are used as the basis of equivalence testing, modified simulations with single-element translations may not be considered equivalent to the canonical method (Figure 8).
Figure 7. Illustration of an equivalence region created using the criterion from Section 7.2. The canonical data point, shown by a red circle, is at the edge of the region. This prevents the modified simulation, whose state (shown by a blue square) differs from the canonical point by only one healthy cell, from being declared equivalent to the canonical simulation. Data points used to create the equivalence region are represented by hollow circles.

However, such simulations should intuitively be considered equivalent, because a single-element translation in a large simulated tissue does not seem to be a significant difference. If equivalence regions are constructed for tissue blocks rather than single elements, small translations are less likely to affect equivalence, because translations are likely to have a smaller effect on total block populations.
than on populations of single grid elements (Figure 9). The neighborhood size can be changed to affect the degree of translation allowed. In the most extreme case, setting a neighborhood size equal to the total grid size makes the spatial distribution of cells irrelevant for determining equivalence, which is based only on total tissue-wide cell populations. This modification would be useful in applications for which global cell populations (and not their spatial distributions) are the only result of interest.

2.4.2. Explaining the Concentric Circle Pattern of Non-Equivalence. A possible explanation of the concentric circle pattern of non-equivalent tissue blocks is as follows. Update suppression first occurs near the simulation edges, where nutrient concentration change is lowest. This area of the tissue contains healthy cells with population density at the simulation-enforced maximum. Nutrient concentrations would allow the healthy cells to increase their population above this maximum, so the small error induced by update suppression has no effect on healthy cell populations. Nutrient diffusion spreads the error to the tissue center, where the growing tumor is located. Here, tumor cells are growing dependent on local nutrient concentrations; concentration error changes cell growth rates, causing the simulation to be declared non-equivalent at the tissue center. As the rapidly-dividing fringe of tumor cells expands outward, so does the non-equivalent region. As cell growth decreases in the tissue center, the modified and canonical results converge to the same stable condition, restoring equivalence. Another instability-causing event (such as massive tumor cell death) eventually occurs, leading to a second central
Figure 8. Example of how single-element translations in a 2-by-2 tissue can result in simulation non-equivalence. In (A), cell populations for the canonical simulation, two reasonable tissue-wide result states, and the modified simulation are shown. The modified simulation’s state is the canonical state translated one unit left. In (B), the equivalence region for the upper left tissue location is shown in red. The modified data point, in blue, is not within the equivalence region, causing the modified simulation to be declared non-equivalent. In contrast, Figure 9 shows that, if tissue blocks are used as the basis for equivalence, the translation in this Figure does not result in non-equivalence.
2. CRITERION BASED ON MEASUREMENTS OF TISSUE START STATE

Figure 9. Example of how using blocks instead of single grid elements to construct equivalence regions can prevent single-element translations from causing non-equivalence. (A) is a 2-by-2 block view of the canonical simulation, modified simulation, and reasonable tissue-wide result states from Figure 8 (A). (B) shows the equivalence region for the 2-by-2 neighborhood. The modified simulation’s data point (in blue) is inside the equivalence region, so it is declared equivalent to the canonical method.

region of instability and producing the observed pattern of concentric circles of non-equivalence.
2.4.3. *Problems with Equivalence Criterion.* Criterion 2 has two characteristics that may lead to undesirable results. First, it is possible for modified simulations that are nearly identical to the canonical simulation to be declared non-equivalent. This could occur whenever the canonical simulation’s point is a vertex of the equivalence region, because single-cell deviations could move the modified simulation outside the equivalence region. To avoid this situation, equivalence regions must be constructed so that the canonical point is always centrally located inside the regions, allowing modified simulations that deviate slightly from the canonical results to be declared equivalent to the canonical method. The criterion described in Section 7.3 has this property.

Secondly, the number $z$ of computed reasonable tissue-wide result states may be too low. The convex hull of the $z$ tissue-wide result states is taken to estimate the shape of the entire set of tissue-wide result states, computed and uncomputed. This estimate will only be accurate under two conditions: The random result states chosen for computation must be extreme points of complete set’s three-dimensional representation, and the complete set of reasonable tissue-wide result states must be convex. Because the complete set contains a very high number of points, and only 25 are plotted to form the convex hull, the plotted points are unlikely to be extreme points, in which case the convex hull would not encompass all points in the complete set. This could cause performance-enhanced simulations that lie outside the convex hull to be declared non-equivalent even if their states in all tissue blocks are reasonable result states. Increasing the number $z$ of points
used to compute the convex hull may therefore increase the likelihood of finding equivalent simulations.

3. Criterion Based on Distance from Canonical Simulation

The third equivalence criterion considers a modified simulation $M$ to be equivalent to a canonical method $C$ with respect to start state $S$ and timespan $t_f$ if and only if the following is true:

**Criterion 3.** *For each 9-by-9 tissue block at each timestep up to $t_f$, $M$’s cell populations are within $e\%$ of $C$’s populations.*

Like the criterion from Section 7.2, this criterion is based on the inaccuracy of biological measurements and can be implemented by plotting cell population data for the modified simulation and testing for inclusion in a polyhedron. Unlike the criterion from Section 7.2, the polyhedra are based only on the original start state $S$, not on computations of reasonable result states.

3.1. Theoretical Justification. This criterion considers two simulations to be equivalent at every timestep if their results could not be distinguished by measurements with an error range of $e\%$. That is, if the modified cell populations are within $e\%$ of the canonical cell populations, then measurements would not be able to distinguish between a tissue sample growing according to the canonical predictions and one growing according to the modified predictions. Such simulations are considered equivalent by Criterion 3.
3.2. Implementation. This criterion is implemented in a straightforward manner: The canonical method $C$ and the modified method $M$ are both run for $t_f$ timesteps on the start state $S$. The normal, tumor, and necrotic cell population histories of $M$ and $C$ are then compared. If $M$’s cell populations are within $e\%$ of $C$’s populations in all 9-by-9 tissue blocks for all timesteps from 1 to $t_f$, then $M$ is considered equivalent to $C$. (The parameter $e$ is set to 10 in the current implementation \[5\]).

This implementation can be described as a polyhedron inclusion test like that used in Criterion 2. For each tissue block $q$ at each timestep $t$, the cell population of the canonical simulation is plotted as a point $p = (n, u, r)$ in three-dimensional space with dimensions for normal, tumor, and necrotic cell populations. A 3-box polyhedron centered on $p$ is then defined, with side lengths $2en$, $2eu$, and $2er$. If the modified simulation’s cell population point $p = (n', u', r')$ in block $q$ at time $t$ is within the 3-box, the modified simulation is considered equivalent to the canonical method in block $q$ at time $t$.

3.3. Equivalence Between Performance Enhancement and Canonical Method. As with Criteria 1 and 2, Criterion 3 declares performance-enhanced simulations with very low threshold-settings ($\varepsilon_O, \varepsilon_G, \varepsilon_H$) to be non-equivalent to the canonical simulation. In one case, it declares a performance-enhanced simulation non-equivalent due to a population difference of two cells in a 9-by-9 neighborhood. Healthy cell populations in the neighborhood are low, as nearby tumor growth is starving the healthy cells of nutrients. The canonical simulation
has 17 healthy cells in the neighborhood, and the modified simulation has 15 cells. The criterion’s error range $e$ is 10% [5], so the equivalence region’s limits are at $16 = (17 - \lfloor 0.1(17) \rfloor)$ and $18 = (17 + \lceil 0.1(17) \rceil)$ cells. The modified simulation is therefore declared non-equivalent, despite having a population only two cells different from the canonical simulation in an 81-grid element block.

3.4. Discussion. Unlike the criteria from Section 7.2, the 3-box equivalence regions created by the criterion discussed in this section are centered on the canonical simulation’s state $p$. In contrast, Figure 7 shows that the canonical simulation’s state may be on the edge of equivalence regions generated by the criteria from Section 7.2. Thus, the criterion discussed in this section would seem to avoid the problem mentioned in Section 7.2.4 and demonstrated by Figure 7, in which modified simulations that are nearly identical to the canonical simulation are declared non-equivalent. However, the results in the previous section show that even small differences between the modified and canonical simulations can still prevent equivalence when total cell populations are low, despite the central location of the canonical state within the equivalence region. In Chapter 8, some possible means of addressing this issue are discussed.

4. Non-Equivalence of Performance-Enhanced Simulations

The criteria described in this chapter can be used to find the most efficient performance-enhanced simulation whose results can be considered equivalent in some precisely-defined way to those of the canonical simulation method. In fact,
the performance-enhanced simulation was not considered equivalent to the canonical method using any of the steady-state threshold settings tested during this work; this is true for all three of the equivalence criteria discussed.

To find an equivalent, efficient performance-enhanced simulation, the performance enhancement’s steady-state thresholds (see Chapters 5 and 6) are varied to change the number of grid elements that are declared steady and labeled for update suppression. Lowering the thresholds decreases the number of steady areas, providing less of a performance increase but producing results more similar to those of the canonical method. Raising the thresholds improves performance but is likely to produce greater differences in results.

No equivalent performance-enhanced simulations have yet been found: Even simulations with thresholds too low to increase efficiency are not considered equivalent to the canonical method by any of the discussed criteria. There are several explanations for this difficulty. The first explanation is simulation instability: Update suppression produces increasing differences between performance-enhanced and canonical results, preventing equivalence. An alternative explanation is that the proper setting of the steady state threshold values $\varepsilon_O$, $\varepsilon_G$, and $\varepsilon_H$ has not yet been tested. Finally, the equivalence criteria may not provide adequate definitions of simulation equivalence. These three explanations are discussed below.
4. Non-Equivalence of Performance-Enhanced Simulations

4.1. Simulation Instability.

4.1.1. Reasons for Simulation Instability. Experiments have shown that the simulation is unstable: Small differences in nutrient concentrations produce differences in cell population that increase as time passes. As shown in Figure 1, update suppression for 5 timesteps in one 3-by-3 block in a 54-by-54 tissue causes cell populations in many grid elements to deviate increasingly far from the canonical populations as time passes. Furthermore, as shown in Figures 2 and 3, the population deviation spreads to grid elements increasingly far from the original location of update suppression. This indicates that diffusion causes nutrient concentration difference to spread throughout the tissue from its point of origin, a tissue block where updates are suppressed.

Nutrient concentration differences between the performance enhancement and the canonical simulation contribute to instability by inducing cell population differences. Due to the discrete nature of cell populations, nutrient concentrations can vary within a certain range without affecting cell populations. In some grid elements, nutrient concentrations in the canonical simulation may be in the middle of this range, allowing maximum nutrient concentration variation in a performance-enhanced simulation without affecting cell populations. In other locations, nutrient concentrations in the canonical simulation may be near the range’s limits, so that a small concentration difference in a performance-enhanced simulation could change cell populations. As a result, tissue-wide nutrient concentration error may affect cell populations in some locations but not in others, and may not affect
cell populations at all for a number of timesteps. This may explain why, in Figure 1, cell population error does not appear until approximately 150 timesteps after updates are first suppressed.

In addition, simulation instability is caused by changes in cell population. Cells that die in a performance-enhanced simulation but survive in the canonical simulation cause differences in nutrient consumption between the two simulations, introducing a nutrient concentration difference that spreads via diffusion to adjacent grid elements; cells that reproduce in the canonical simulation but remain inactive in the performance-enhanced simulation have similar effects. Through these mechanisms, nutrient concentration error seems to set off a positive feedback loop by which tissue-wide concentration and population errors steadily increase.

Simulation instability may also be exacerbated by the use of determinism rather than stochasticity in the simulation. Biological systems are best modeled stochastically, although deterministic approaches may suffice when molecular concentrations are high [20]. The simulation’s use of determinism means that arbitrarily small changes in nutrient concentration have exact effects in all tissue locations. In contrast, if diffusion were modeled as a molecular random walk rather than as a deterministic averaging function, the stochasticity-induced noise might absorb small changes in nutrient concentration before they spread throughout the entire grid. Use of stochasticity in other processes, such as cell growth, could have a similar damping effect, possibly increasing simulation stability.
4.1.2. Biological Realism of Instability. The simulation’s instability may not be biologically realistic. Living organisms employ negative feedback loops in order to maintain stable internal conditions. A more comprehensive simulation that includes biological stability mechanisms might be less unstable than the current simulation.

Specifically, nutrient concentrations might be less stable in the simulation than in actual tissue. The simulation models a thin slice of tissue that is treated as being two-dimensional. Nutrient concentration stability is provided by grid elements with constant concentrations: blood vessels and (when fixed boundary conditions are used) the simulation boundary. In the current implementation, blood vessels are single grid elements spaced evenly throughout the tissue. The locations farthest from blood vessels and boundaries may be most susceptible to rapid nutrient depletion because of the long distance to the nearest unlimited nutrient source. However, in biological tissue, nutrients would enter the tissue slice from above and below, not just from the horizontal boundaries. When this three-dimensional nutrient diffusion is considered, all locations are a maximum of one grid element away from an unlimited source of nutrients, allowing nutrient concentrations to remain more stable tissue-wide. The current simulation’s implementation of diffusion as a two-dimensional process may be a partial cause of the observed instability.

4.2. Calibration of Steady-State Thresholds. As explained above, the performance enhancement’s steady-state threshold settings (see Chapter 5) were
varied in an attempt to find a simulation that is more efficient than the canonical method and whose results are considered equivalent to the canonical results by one or more of the equivalence criteria from this chapter. No such simulation has been found, but this does not mean that one does not exist. There may be an as yet untested setting of the steady-state thresholds \((\varepsilon_O, \varepsilon_G, \varepsilon_H)\) that results in such a simulation. To find the correct threshold settings, experiments could be run to isolate the effects of each individual threshold \((\varepsilon_O, \varepsilon_G, \varepsilon_H)\) on simulation equivalence and efficiency. Once the individual effects of the three thresholds are understood, determination of the correct setting may be an easier task.

4.3. Problems with Equivalence Criteria. The discussed equivalence criteria may be excessively strict. As demonstrated in Sections 7.2.3 and 7.3.3, small cell population differences can prevent a modified simulation from being declared equivalent to a canonical simulation. It seems that such small deviations in cell population should not be sufficient cause to declare a modified simulation non-equivalent. For example, if a modified simulation method produces tumor shape and expansion rates identical to those of the canonical simulation, but with a population difference of several cells in a few grid elements, the modified method could be declared non-equivalent by any of the criteria. However, the modified method could still provide useful predictions of tumor growth despite the cell population differences, and the efficiency benefits might outweigh the difference in results. Thus, the equivalence criteria might be considered too restrictive in some cases, and development of other, less strict criteria could be useful.
The cell populations of simulations with similar but distinct starting conditions seem to diverge (i.e., become increasingly dissimilar) during certain timespans and converge (i.e., become more similar) during others. Cell population divergence seems to occur during times of rapid tissue change, such as invasion of healthy tissue by tumor cells, or formation of a necrotic core inside a tumor. Populations seem to converge during times of tissue stability. This phenomenon can be seen by observing reasonable result states over time in an equivalence region of Criterion 2. As described in Section 7.2, an equivalence region’s reasonable tissue-wide result states are simulations run on similar starting conditions. During times of tissue stability, the tissue-wide result states are similar or identical, and the equivalence region is therefore small, as in the two right panels of Figure 4. During times of instability, the tissue-wide result states are less similar, producing a larger equivalence region, as in the left panels of Figure 4. This pattern indicates that similar simulations (such as a performance-enhanced simulation and the canonical method) produce similar results at most timesteps, but deviate during the rapid transitions between stable tissue states.

The deviation of cell populations in similar simulations during rapid tissue change may be responsible for the difficulty in finding performance-enhanced simulations that are considered equivalent to the canonical method by the criteria discussed in this chapter. These criteria place equal weight on all simulation timesteps—the simulation being tested for equivalence must provide similar results to the canonical method at all times. For some applications, the rapid
transitions between tissue conditions may be less important than the actual sequence of stable tissue conditions, making the criteria discussed in this paper too stringent. For example, if the simulation goal was to determine the necrotic core’s initial diameter, accurate simulation of the initial core formation event, in which massive tumor cell death occurs over several timesteps, would be less important than accurate simulation of the stable tissue state that follows the formation event.

Modification of the equivalence criteria to emphasize accuracy during tissue stability over accuracy during rapid transitions would be reasonable for such applications. The modification could be realized by developing a quantitative measure of tissue stability and assigning weights to timesteps based on the current level of tissue stability, with times of unstable tissue receiving lower weights than times of stable tissue. If the simulation being tested for equivalence was not within an equivalence region at time $t$, a penalty would be assigned in proportion to $t$’s weight. The tested simulation would be considered equivalent to the canonical method if and only if the total of all assigned penalties was below a threshold.

Finally, there may be no equivalent performance-enhanced simulations with the tissue size used in this paper. Larger tissues may have more stable, tumor-free areas where updates can be suppressed without significantly affecting conditions in the unstable tissue center. The maximum tissue size in this investigation was limited by memory; the methods demonstrated here may be more useful with the larger tissue sizes possible in more powerful computational environments.
CHAPTER 8

Conclusion and Future Work

1. Conclusion

This paper describes a performance enhancement for a cellular automaton-based tumor simulation and illustrates a method for comparing its results to those of the canonical method. Suppressing updates to steady-state areas improves performance but affects results more than anticipated, revealing simulation instability. In order to judge whether the results of performance-enhanced simulations (Chapter 6, [1]) could be considered equivalent to those of the canonical method (Chapter 3, [11]), biologically-motivated criteria for simulation equivalence were investigated. Performance-enhanced simulations with results nearly identical to those of the canonical simulation were considered non-equivalent by the criteria, indicating the need for development of alternate criteria.

2. Future Work

This section discusses several opportunities for further investigation pertaining to the work described in this paper. The computational efficiency of the hierarchical data structure for tracking steady state areas (Section 6.1) could be optimized through the use of bitwise operations and a system-oriented programming language, such as C [1]. Sources of simulation instability could be investigated and
eliminated by incorporating biologically realistic feedback loops and gene regulatory networks into the simulation. The performance enhancement (Chapter 6) could be adjusted to induce less cell population error. Finally, additional performance enhancements could be investigated.

2.1. Increasing Data Structure Efficiency with Bitwise Operations.

As explained in Section 6.1, the performance enhancement uses a hierarchical data structure to decrease the required number of matrix accesses when tracking the locations of steady-state areas \([1]\). For each nutrient \(x\) in an \(N\)-by-\(N\) simulated tissue, an \(N\)-by-\(N\) matrix \(S_x\) is stored, elements in \(S_x\) correspond to grid elements and label them as steady-state (zero) or unstable (one). An \(\frac{N}{3}\)-by-\(\frac{N}{3}\) matrix \(S\beta_x\) represents a low-resolution view of \(S_x\); elements of \(S\beta_x\) represent 3-by-3 neighborhoods in \(S_x\). Elements of \(S\beta_x\) are zero if and only if the corresponding neighborhood of \(S_x\) is entirely zero. Similarly, an \(\frac{N}{9}\)-by-\(\frac{N}{9}\) matrix \(S9_x\) represents 3-by-3 neighborhoods of \(S\beta_x\) and 9-by-9 neighborhoods of \(S_x\). When using the data structure to locate unstable grid elements, the hierarchical approach potentially requires fewer data structure array accesses than if \(S_x\) were used alone (see Section 6.1).

When grid elements are reassigned as steady-state or unstable, the corresponding elements of the data structure must be reset. In the current MATLAB implementation, the matrices \(S_x\), \(S\beta_x\), and \(S9_x\) are set in a straightforward manner. \(S_x\) is set based on nutrient concentration changes as described in Section 6.1. Each 3-by-3 neighborhood in \(S_x\) is searched for non-zero elements using MATLAB’s
built-in matrix search function; if none are found, the corresponding element of $S3_x$ is set to zero. $S9_x$ is then set using $S3_x$ in an analogous manner.

The data structure’s performance could potentially be improved by using bitwise operations to set the $S3_x$ and $S9_x$ matrices [1]. The nine-element neighborhoods of $S_x$ corresponding to single grid elements of $S3_x$ can be interpreted as eight-bit numbers if the center element is disregarded. These numbers can be tested for equality with zero (a fast bitwise operation) in order to set the corresponding element of $S3_x$. Similarly, $S9_x$ can be set by treating the eight outer elements of each 3-by-3 neighborhood of $S3_x$ as an eight-bit number and checking for equality with zero. If correctly implemented, this method could be more computationally efficient than iterating through the nine elements in search of nonzero values.

This method of setting the hierarchical data structure ignores the central element of the three-by-three neighborhoods, potentially leading to incorrect settings. Figure 1 provides an example: If the central grid element in a neighborhood of $S_x$ is 1, and the eight surrounding elements are 0, the eight-bit interpretation will be equal to zero, and the corresponding element of $S3_x$ will incorrectly be set to zero. This would lead to suppression of updates to the corresponding tissue location, when in fact it should have been updated due to the central unstable element.

Testing would be necessary to determine how often incorrect data structure settings of the type shown in Figure 1 occur in simulations. Adjustments could be made to eliminate the possibility of incorrect settings, but this may not be
as tissue conditions resulting in incorrect settings may occur rarely, due to the properties of simulation instability. As argued in Section 6.3.1, tumor cells are the agents of change in the simulation. Tissue instability originates at locations containing tumor cells and spreads outward as time passes. Thus, if a 3-by-3 neighborhood contains a single unstable element, it is unlikely to be the central element, because the instability has likely spread to the neighborhood from a neighboring location. Thus, the first appearance of instability in a neighborhood is more likely to occur in an edge element than in the center element. Spontaneous appearance of instability in the central element of an otherwise stable neighborhood is only likely if the central element contains an instability-promoting agent
(i.e., a tumor cell). However, this situation is also unlikely to occur, because tumor cells remain confined to the main tumor mass, which is located in a highly unstable tissue region. Therefore, any 3-by-3 neighborhoods containing tumor cells would likely already be entirely unstable; they would not contain a stable outer area and an unstable central element. Thus, tissue conditions that produce incorrect settings of the data structure may be relatively rare; if not, the data structure could be adjusted to eliminate the possibility of incorrect settings.

2.2. Investigating Simulation Instability. As illustrated in Section 7.1.3, small modifications to the simulation method induce cell population differences that increase over time. The simulation rules and parameters could be investigated to find and remove the sources of this instability, which may not be biologically realistic (Section 7.4.1.2). First, a definition and quantitative measure of instability could be developed: Instability could be defined as change in tissue state and quantified by measuring the changes in cell populations and nutrient concentrations over several timesteps. With this method, tissue areas undergoing events such as invasion by tumor cells or necrotic core formation would be considered highly unstable, due to the large cell population changes that occur during those events. Unchanging areas, such as healthy tissue far from the growing tumor, would be considered stable.

Simulation rules could be modified, added, or eliminated to determine their individual effects on quantitative measures of tissue instability. Rules found to
decrease instability could then be implemented to produce a more stable, biologi-
cally realistic simulation. It is possible that modification of simulation rules might
not be sufficient to decrease instability; more comprehensive simulation of biolog-
ical processes may be necessary. Omission of important processes such as tumor
cell quiescence (the cessation of reproduction by tumor cells) or centrally-directed
tumor cell motion might cause or increase instability, because those processes may
serve to maintain homeostasis (stable internal conditions) in living organisms. For
example, tumors in the current simulation method (Chapter 3, [11]) grow until
they invade most of the simulated tissue, and they stop growing only when they
near the simulation edges, causing tissue-wide instability. This is biologically un-
realistic, because tumors of the type modeled by the simulation, which do not
induce development of new blood vessels, tend not to grow beyond a maximum
radius [24]. If the simulation were modified to limit tumor growth, outer tis-
sue areas would not be invaded by tumor cells, and tissue-wide instability would
therefore decrease.

Several biologically-motivated mechanisms for limiting tumor growth could
be added to the simulation. The current implementation (Chapter 3, [11]) com-
putes cell reproduction as a function of nutrient concentration: Tumor cells stop
dividing when nutrient concentrations are too low to sustain further division.
However, [6] states that low nutrient concentrations cannot be the only cause of
tumor cell quiescence in vivo, because nutrient concentrations must drop to lev-
els that would never occur in living organisms before cells stop dividing in vitro;
thus, additional factors, such as signaling molecules, must be involved in vivo. In addition, the authors of another cellular automaton-based tumor simulation [16] suggest that inward tumor cell movement might be responsible for the cessation of pre-angiogenic tumor growth. Expanding the simulation to include signaling molecules and inward cell movement might slow or stop tumor growth beyond a certain radius, improving biological realism and reducing instability.

2.3. Tuning the Performance Enhancement for Greater Accuracy.

The performance enhancement in Chapter 6 [1] could be adjusted for improved accuracy. The current implementation introduces differences in nutrient concentrations compared to the canonical method by suppressing updates to grid elements, causing nutrient concentrations to remain constant when they would be changing in the canonical method. In an alternative implementation, updates to steady areas could be simplified rather than suppressed completely. Rather than running the complete nutrient diffusion routine in steady-state grid elements, the last known local nutrient concentration change could be reapplied at each timestep. The implicit assumption in this implementation is that, in steady areas, the rate of nutrient concentration change is likely to remain relatively constant. In contrast, the current implementation implicitly assumes that nutrient concentrations themselves remain constant in steady areas. The former assumption is likely a more accurate approximation of tissue behavior than the latter, so the alternative implementation may decrease nutrient concentration differences if the rate of nutrient concentration change remains stable over the course of several timesteps.
This implementation would require more computation than the current version in which updates are completely suppressed. However, it would still be more efficient than the canonical method, because complete diffusion updates require matrix multiplication, a costly operation, whereas reapplying the last known concentration change requires array addition.

In another possible implementation, updates to steady-state grid elements would be completely suppressed, as in the current implementation described in Chapter 6. However, steady-state elements would be updated every $k$ timesteps (where $k$ is a specific, fixed value), and the nutrient concentration change $c$ between the current timestep $t$ and the timestep of last update $u$ would be computed for each steady-state grid element. The nutrient concentration in each steady-state element would then be changed by $c(t - u)$. This is done to compensate for the possibility that, had updates not been suppressed over timespan $u..t$, nutrient concentrations in a steady-state element may have changed by $c$ at each timestep between $u$ and $t$. This total nutrient concentration change is therefore retroactively applied at timestep $t$.

**2.4. Simulating Protein Regulatory Networks.** The current simulation rules make some simplifications of biological knowledge in order to ease implementation and improve efficiency. For example, no modeling of individual cells with independent behaviors is incorporated: Cells are treated as local populations that collectively consume nutrients, reproduce, and die [11].
In the simulation, cell reproduction is a function of local nutrient concentrations; growth of actual biological tumors is affected by protein regulatory networks, which could be incorporated into the simulation to improve realism. Protein regulatory networks are collections of interacting cellular proteins that translate external stimuli to cellular responses. External factors, such as signaling molecules, initiate a cascade of chemical modification to various proteins, eventually leading to the modification of proteins involved in gene expression and causing cellular responses such as reproduction or death.

The simulation from [11] simplifies the phenomenon of quiescence. In the current implementation, cells enter quiescence when nutrient concentrations limit energy production to the point that all available energy must be devoted to critical life processes, leaving none for reproduction [11]. In contrast, biological experiments [6] show that nutrient concentration affect pre-angiogenic tumor growth, it cannot be solely responsible for cell quiescence: Quiescence has been observed in tumors with nutrient concentrations high enough to allow reproduction, indicating that other, unknown factors must be responsible [22].

Rather than modeling cell reproduction directly as a function of nutrient concentrations, reproductive control by protein regulatory networks could be simulated. The multiscale cellular automaton-based tumor model described in [16] takes this approach; the network model is illustrated in Figure 2. Growth factor and inhibitory factor concentrations are the input to the regulatory network. A quantitative factor level is computed from the two concentrations; factor level
increases with growth factor concentration and decreases with inhibitory factor concentration. Each protein has an individual, specific threshold $\theta$; if the factor level is greater than $\theta$, the protein becomes primed. Protein state (on or off) is then decided with the following rules:

1. If all links to a primed protein are inhibitory, and all proteins that inhibit primed protein expression are off, then the primed protein is turned on.
2. If all links to a primed protein are stimulatory, and all proteins that enhance primed protein expression are on, then the primed protein is turned on.
3. Primed proteins are turned off in any other situation.
4. Non-primed proteins are always turned off.

In this way, the network maps growth and inhibitory factor concentrations to expression of the final protein E2F (See Figure 2). Cells in which E2F is on divide; cells in which it is off enter quiescence. This type of regulatory network could be incorporated into the simulation discussed in this paper, providing a more complex, biologically realistic model of tumor growth.

Simulation of genetic mutations could be incorporated into the current model, as well. Cell genomes are not completely stable: healthy cells become cancer cells through genetic mutations. Existing tumor cells can also experience mutations, possibly altering their characteristics. For example, [4] cites evidence that hypoxia causes genetic instability in tumor cells: The resulting mutations may lead to effects such as increased tumor malignancy. Furthermore, hypoxia modifies
Figure 2. An illustration of the gene regulatory network model from [16]. Each box represents a protein. An arrow from one protein (call it protein A) to another (call it B) indicates that A stimulates the expression of B. A line between A and B ending in a horizontal segment indicates that A inhibits the expression of B. From [16].

the behavior of cell regulatory networks, stimulating the transcription of proteins involved in glycolysis, cell survival, glucose transport, and angiogenesis (the induction by tumor cells of blood vessel growth to increase nutrient supply) [4]. Incorporating the genetic and regulatory effects of hypoxia and other extracellular conditions could result in a more biologically realistic simulation.

Additional criteria for simulation equivalence could be defined in order to account for these additional simulated components. If cell populations were the only simulation result of interest, the criteria described in this paper could still
be used to judge equivalence between multiple simulation methods incorporating protein networks and genetic mutations. If, however, protein network states or mutation events were important results, equivalence criteria that explicitly take these factors into account would be more useful.
Bibliography