A THREE-DIMENSIONAL COMPUTATIONAL MODEL FOR THE GROWTH OF MULTICELLULAR TISSUES AND ITS PARALLEL IMPLEMENTATION ON A CLUSTER

by

Lenny C. K. Tang

B. Sc., Simon Fraser University, 2003

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
in the School
of
Interactive Arts and Technology

SIMON FRASER UNIVERSITY
Summer 2007

APPROVAL

Name: Lenny C. K. Tang

Degree: Master of Science

Title of thesis: A Three-Dimensional Computational Model for the Growth of Multicellular Tissues and Its Parallel Implementation on a Cluster

Examining Committee:

Dr. Christopher D. Shaw
Associate Professor, Graduate Chair
School of Interactive Arts and Technology

Dr. Belgacem Ben Youssef
Assistant Professor, Senior Supervisor
School of Interactive Arts and Technology

Dr. Kay C. Wiese
Assistant Professor, Supervisor
Computing Science

Dr. Vive Kumar
Adjunct Professor, External Examiner
School of Interactive Arts and Technology

Date Approved: May 10, 2007

Abstract

We report the development of a computational model for the growth of multicellular tissues based on cellular automata to study the tissue growth rates and population dynamics of multiple populations of proliferating and migrating cells. Cell migration is modeled using a Markov chain approach and each population of cells has its own division and motion characteristics based on experimental data. The extended model contains a number of parameters that permits the study and analysis of cell population dynamics. This allows us to explore their effects on the overall tissue growth rate and the frequency of cell-cell interactions due to collision and aggregation. In addition to a sequential implementation, we developed a parallel algorithm and implemented it on a Beowulf Cluster using the Message Passing Interface. We present the sequential and parallel simulation results and analyze the performance of the parallel algorithm in terms of speedup and efficiency.

Keywords: cellular automata; tissue growth; three-dimensional model; parallel algorithm

To my loving parents...

"The best part of falling is getting back up again."

— David Belle

Acknowledgments

This thesis would not have been possible without the assistance of many people. My deepest gratitude goes to Dr. Belgacem Ben Youssef for his constant support, patience, and wisdom throughout my research. Dr. Ben Youssef's tireless efforts and precision have helped shape my thesis into what it is today. I would also like to thank Dr. Kay Wiese for introducing me to academic research, encouraging me to pursue a graduate degree and accepting to be on my supervisory committee; Dr. Vive Kumar for his guidance during my undergraduate studies and generously serving as my external examiner.

I also would like to acknowledge the financial support received from the Natural Sciences and Engineering Research Council (NSERC) of Canada and the School of Interactive Arts & Technology at Simon Fraser University in the form of Research Assistantships and Teaching Assistantships, respectively. My appreciation goes to all those who constantly dealt with the temperament of our Nebula Beowulf Cluster and kept the machines up and running, especially: Patrick Lougheed, Mark Deepwell, Justin Thomas, and Gordon Pritchard.

I must thank my colleagues and friends who have guided me and kept me entertained in the past years: Herbert Tsang, for all those refreshing talks and afternoon jaunts to nearby fast food restaurants; Andrew Park, for the friendship expressed with tasty chocolates and a kindness rarely seen this day and age; Eddie Hou, for faithfully following me from school to school to school; Wen-Ling Ko, for her companionship and being that caring ear when all the world was dark; Coco Jiang, for being funny; and Huaxin Wei, for everything.

I would like to express my appreciation to my family who supported me for all these years: to my brother Clarence who has always been there for me and to my parents Elaine and Nelson Tang for providing me an environment and luxury that allowed me to study and grow. The ceaseless encouragement and support from them throughout the years have meant everything to me.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Each year, millions of surgical procedures are performed to relieve patients who are affected by tissue loss, due to burns and injuries, or organ failure. Operations treating patients using tissue reconstruction and organ transplantation have been highly successful. However, the number of patients treated by these therapies is small due to the limited number of donors available. The primary focus of tissue engineering is the growth of three-dimensional tissues with proper structure and function. Tissue engineers combine knowledge from the areas of biochemistry, medical sciences, and engineering to develop bioartificial tissue substitutes or to induce tissue remodelling in order to repair, replace, or enhance tissue functions [33, 46].

Natural tissues are multicellular and have a specific three-dimensional architecture. This structure is supported by the extracellular matrix (ECM). The ECM often has the form of a three-dimensional network of cross-linked protein strands. In addition to determining the mechanical properties of a tissue, the ECM plays many important roles in tissue development. Biochemical and biophysical signals from the ECM modulate fundamental cellular activities, including adhesion, migration, proliferation, differentiation, and programmed cell death [57]. Recent studies of biomaterials have provided ways to manipulate some of these cellular activities through the use of targeted fabrication [22] or modification of bioartificial scaffolds [42, 26].

Tissue engineers grow bioartifical tissue substitutes by reproducing the structural components of naturally grown tissues. However, tissue growth is a complex process affected by

many contributing factors such as the type of cells, initial seeding densities, spatial distribution of the seed cells and the culture conditions [52]. The dynamic process for generating cellularized tissue substitutes can be described as follows:

- A small tissue sample is harvested from the patient or donor.
- Cells are isolated, cultured and seeded into a three-dimensional scaffold with the proper structure and surface properties. The scaffold may contain growth factors to induce and maintain the proper differentiated cellular function.
- The cells migrate in all directions and proliferate to populate the scaffold. Cell migration speeds and proliferation rates are controlled by the surface properties and morphology of the scaffold. Bioactive agents may also be used to regulate cell migration and proliferation.
- The bioartificial tissue substitute is implanted in the patient.

Wound healing is the process by which cells surrounding the wound proliferate into the wound area and, over a period of time completely cover it. Tissue engineering is used in wound healing. A biocompatible matrix or scaffold is utilized to fill the wound. This scaffold induces neighbouring cells to migrate into it, proliferate and produce their own extra cellular matrix. This process forms a new tissue with the right differentiation and, thus, heals the wound.

Scaffold properties, cell activities like adhesion or migration, and external stimuli that modulate cellular functions are among the many factors that affect the growth rate of tissues. Hence, the development of bio-artificial tissue substitutes involves extensive and time-consuming experimentation. The availability of computational models with predictive abilities will greatly speed up progress in this area.

This research focuses on the development of a computational model to simulate the growth of three-dimensional tissues consisting of more than one cell type. Specifically, this model can be used to study how the overall tissue growth rate is affected by:

- cell migration speed;
- the initial density of the seed cells; and
- their spatial distribution.

The success of models that describe the dynamic behaviour of cell populations, however, will depend on our ability to accurately describe the migration and proliferation of different cell types in various environments. Such data are obtained in experiments in simpler and therefore, easier to control systems.

1.2 Modeling and Simulation Approaches

Modeling is the process of establishing interrelationships between important entities of a system, where models are represented in terms of goals, performance criteria and constraints. Computer simulation is the discipline of designing a model of an actual or theoretical physical system, executing the model on a computer, and analyzing the execution output. Simulation embodies the principle of learning by doing. To understand reality, nature and all of their complexities, we must build artificial objects and dynamically act out roles with them [34].

Simulation is a tightly coupled and iterative three component process composed of model design, model execution, and execution analysis. The first step to building a simulation model of a real system is to gather data associated with that system. From the data and knowledge of past experiments with similar systems, we formulate the model. Models must be converted to algorithms to run on a digital computer serially. Parallel simulation methods, while generally applicable and far reaching, are used on parallel machine architectures.

In order to develop a model to study a real-world system scientifically, a set of assumptions are required. These assumptions, which usually take the form of mathematical or logical relationships, constitute a model that is employed to gain some understanding of the behaviour of the corresponding system. If the relationships describing the model are simple enough, an analytical solution can be obtained by using mathematical modeling. However, most real-world systems are too complex to allow analytical evaluation, thus, enforcing the simulation approach. In this methodology, a computer is used to evaluate a model numerically, and gathered data is used to estimate the characteristics of the model. In particular, modeling deals primarily with the relationships between real systems and models, while simulation refers to the relationships between computers and models.

There are two primary strategies to evaluate the correctness of a model. They are verification and validation [34].

Definition 1.1 Verification. Checking that the simulation program performs correctly and as intended. This involves checking, for instance, whether the implementation is

free of errors and consistent with the model.

Definition 1.2 Validation. Determining whether the simulation model, correctly implemented, is an accurate representation of the system under study. That is, is the model a sufficient close approximation to reality for the intended application?

Although the concepts of verification and validation are distinct, in practice they may overlap to a considerable extent. When a simulation produces erroneous output, it is not always clear whether this is due to errors in the model, implementation errors, or the use of faulty input data. Model development, verification, and validation should be done hand-in-hand throughout the simulation study. A worthwhile model must generate predictions that are then corroborated by observations of, or experiments with, the real system.

1.3 Cellular Automata Concepts

Cellular automata (CA) were originally introduced by John von Neumann and Stan Ulam as a possible idealization of biological systems with a particular purpose of modeling biological self-reproduction [62]. This approach has been used since then to study a wide variety of physical, chemical, biological, and other complex natural systems.

We consider d-dimensional cellular automata consisting of an array D of lattice cells covering a finite domain. Any cell c is uniquely identified by d integer coordinates (i_1, i_2, ..., i_d), where $1 \leq i_1 \leq N_1$, $1 \leq i_2 \leq N_2$, ..., and $1 \leq i_d \leq N_d$. Let Ω be the set of all computational sites in the cellular space.

Definition 1.3 A cellular automaton satisfies the following properties:

[P-1] Each cell c interacts only with its neighbour cells defined by a neighbourhood relation that associates with the cell c a finite list of neighbour cells $c+v_1, c+v_2, \ldots, c+v_k$. In general, the neighbourhood vector (or neighbourhood index), $V = [v_1, v_2, \ldots, v_k]$, may vary from one cell to another.

[P-2] Each cell can exist in one of a finite number of states. This finite list of states will be listed by Q. In the simplest case of two-state automata, $Q = \{0, 1\}$.

[P-3] Each function $X : \Omega \rightarrow Q$ defining an assignment of states to all cells in the cellular space Ω is called a configuration. Then, x_c is called the state of the cell c under configuration X.

[P-4] For any cell c in the cellular space, there exists a local transition function (or rule) f_c, from Q^k to Q, specifying the state of the cell c at time level $t+1$ as a function of the states of its neighbours at time level t ($x_c^{t+1} = q^{t+1}(c)$) , i.e.,

$$x_c^{t+1} = f_c(x_{c+v1}^t, x_{c+v2}^t, \ldots, x_{c+vk}^t).$$

[P-5] The simultaneous application of the local transition functions (or rules) f_c to all the cells in a cellular space defines a global transition function F which acts on the entire array transforming any configuration X^t to a new configuration X^{t+1} according to

$$X^{t+1} = F(X^t).$$

These properties imply that each cellular automaton is a discrete dynamical system. Starting from an *initial configuration* X^0, the cellular array follows a trajectory of configurations defined by the global transition function F. All the possible configurations x of the cellular automaton define a set Φ whose cardinality can be quite large. For $N_1 = N_2 = N_3 = 5$ and $Q = \{0, 1\}$, for example, Φ contains $2^{5\times5\times5} = 2^{125} \approx 4.2535 \times 10^{37}$ configurations.

We can now define parallel discrete iterations for a cellular automaton as follows:

$$\begin{cases} X^0 & = \text{is given in } \Phi \\ X^{t+1} & = F(X^t) \end{cases}$$

for t = 0, 1, 2, ... or, equivalently

$$\begin{cases} X^0 & = (x_1^0, x_2^0, \ldots, x_N^0) \text{ is given in } \Phi \\ X_f^{t+1} & = f_i(x_1^t, x_2^t, \ldots, x_N^t), \end{cases}$$

for $t = 0, 1, 2, \ldots$ and $i = 1, 2, 3, \ldots, N$. The preceding two equations (or rules) imply that the parallel discrete iterations update the states of all the cells at the same time. A cellular automaton is said to be uniform if the neighbourhood relation and the transition function are the same for every cell. The transition functions of cellular automata need not be algebraic in form and may be rule-based.

Advantages and Drawbacks of Cellular Automata

The use of cellular automata in modeling various systems including biological ones has significant advantages and certain drawbacks. We list a number of advantages below:

- Cellular automata provide a computationally proficient technique for analyzing the collective properties of a network of interconnected cells.
- Models based on cellular automata provide an alternative approach involving discrete coordinates and variables to represent the complex dynamic system.
- The model behaviour is completely specified by a simple operating mechanism in terms of local relations, which is sufficient to support a whole hierarchy of structures and phenomena.
- This simple method of representing complex systems offers the ability to describe all the parameters governing the growth and motion of cells in a computationally efficient way.
- Algorithms based on cellular automata are also suitable for parallel processing.

Using cellular automata is not without its drawbacks. Some problems identified by Wolfram in [62] are listed below:

- In order to closely approximate a continuum of results, it is necessary to process cellular automata using a sufficient number of sites and over a sufficient number of time steps.
- Imperfections (for example, those due to probabilistic rules or heterogeneous processing of the cellular automaton lattice) can have a large effect over time.
- It is not known how to reconstruct the initial state of a cellular automaton from the random patterns of temporal sequences generated after a finite number of time steps.

1.4 Research Objectives and Contributions

This research sets out to extend the functionality of previously developed three-dimensional cellular automata based models for the study of population dynamics of migrating and proliferating mammalian cells. One of the primary goals of this research is to study the competing behaviours when multiple mammalian cell populations are proliferating in the same cellular space. The objective of this model is to be able to compute the tissue growth rate and predict the time to reach volume coverage when the different properties of cell

migration and division are known. With proper experimental validation, such a model has the potential to be a predictive tool.

The model discussed in this thesis is based on an earlier model implemented in *FORTRAN*. Our extended model was implemented using the *C++* programming language. We employed object-oriented programming techniques to enhance the flexibility and extensibility of the implementation. In addition to our sequential algorithm, we implemented a parallel algorithm that runs on a Beowulf Cluster. The performance of the parallel algorithm is analyzed in terms of speedup and efficiency.

Models with predictive capability are a necessary prerequisite for developing systems control strategies for biotechnological processes involving the growth of multicellular tissues. The significance is that this three-dimensional model will have implications in speeding up progress in the area of tissue engineering where the development of bioartificial tissues involves extensive and time-consuming experimentation.

1.5 Overview of the Thesis

This chapter provides a brief introduction to Tissue Engineering, modeling and simulation approaches, and cellular automata. It outlines the need to have a simulation approach to our model and how the use of cellular automata simplifies the simulation approach.

In Chapter 2, we review previous work by discussing the different types of models used to study and analyze the proliferation and migration of anchorage dependant contact-inhibited endothelial cells. Next, Chapter 3 describes the basic processes of cell proliferation, migration, collision and aggregation. A description of our extended three-dimensional model for cell locomotion and division using a discrete-time Markov chain approach and based on cellular automata is also included. The sequential algorithm and its corresponding flowcharts are presented in Chapter 4. In Chapter 5, we discuss the reasons for implementing the algorithm in a parallel computing environment and present the parallel algorithm using a slab decomposition technique.

Chapter 6 presents a subset of the simulation results obtained from both the sequential and parallel implementations of the model for different cell topologies and distributions. Additional simulation results are presented in Appendix A. In Chapter 7, we analyze the performance of the parallel simulations on a Beowulf Cluster. Here, we discuss the speedup and efficiency of our implementation for different processor numbers and cellular array

sizes. We then present our conclusions and future work in Chapter 8. Finally, we provide in Appendix B a glossary of terms used in this research.

Chapter 2

Previous Work

This chapter reviews previous studies related to our area of research. Different types of models used to analyse the proliferation and migration of anchorage-dependent contact-inhibited endothelial cells, a type of mammalian cell, are presented. The description of each model and its limitations are given.

Various modeling approaches have been used to simulate the population dynamics of proliferating cells. The models reviewed in this chapter can be classified as: *deterministic*, *stochastic*, and based on *cellular automata*. Early attempts to model cell population growth were limited to nonmotile cells and the study of contact inhibition phenomena on the proliferation of anchorage-dependant endothelial cells. Endothelial cells are intricately involved in many important pathological and physiological processes [31]. The thin monolayer of endothelial cells, which lines the entire vascular system, regulates the exchange of nutrients and waste products between the blood vessels and surrounding tissues. Endothelial cells create an adaptable life support system by pervading the blood vessels of every region of the body [35]. These models limited growth to two dimensions [64], or to microcarriers [24]. Later models added cell locomotion and extended the modeling to three dimensions. A summary of these models is shown in Table 2.1.

2.1 Deterministic Models

Frame and Hu developed a model based on an empirical approach to describe the effects of contact inhibition on the growth rate using one cell type [18]. They imposed the following three criteria on the model:

Table 2.1: Summary of models developed for predicting the migration and/or proliferation of cells. Note that the cellular automata models are also stochastic.

Author(s)	Type	Main Features	Limitations
Frame and Hu (1988) [18]	Deterministic	• Specific growth rate was given in terms of cell density.	• Cells considered to be equally contact-inhibited.
Cherry and Papoutsakis (1989) [11]	Deterministic	• Included perimeter cell growth into growth rate.	• Assumed colonies to be circular. • No colony mergings.
Lim and Davies (1990) [41]	Stochastic	• Random direction of cell division. • Cells represented as irregular polygons.	• Restricted assumptions on cell-cell interactions. • Cell motion not accounted for.
Ruann, Tsai, and Tsao (1993) [54]	Stochastic	• Density-dependent growth. • Cell motility and change in cell size.	• Cell motion was restricted. • Division simplified.
Zygourakis, Bizios, and Markenscoff (1991) [64]	Cellular Automata	• Contact inhibition during all stages of proliferation.	• Cell motion not included.
Forestell, Milne, and Behie (1992) [16]	Cellular Automata	• Cell growth on micro-carriers.	• Number of cells and their neighbours restricted.
Hawboldt, Kalogerakis, and Behie (1994) [24]	Cellular Automata	• Usable with any cell line or type of micro-carrier.	• No cell motion. • Synchronous growth.
Lee, Kouvroukoglou, McIntire, and Zygourakis (1995) [36]	Cellular Automata	• All essential features of cell motion and division.	• Two-dimensional. • One cell per square.
Ben Youssef (1999) [3]	Cellular Automata	• Three-dimensional. • Motion and division.	• Cell size does not vary.
Kansal, Torquato, Harsh, Chiocca, and Deisboeck (2000) [28]	Cellular Automata	• Studies the growth of tumours •Three-dimensional.	• Simplified cell representation. • No cell migration.
Chang, Gilbert, Eliashberg, and Keasling (2003) [9]	Cellular Automata	• Three-dimensional. • Division and death.	• No cell migration. • No contact inhibition.
Cheng, Ben Youssef, Markenscoff, Zygourakis (2005) [10]	Cellular Automata	• Three-dimensional. • Enables cellular aggregation.	• A single cell type.
Cickovski et al. (2005) [12]	Cellular Automata	• Simulates morphogenesis. • Processes multiple cell types.	• No cell migration.
Our model: Tang and Ben Youssef (2006) [59]	Cellular Automata	• Multiple cell types. • Allows the formation of multicellular aggregates.	• Cell size does not vary.

1. At low cell density, the specific growth rate is constant.

2. At confluence, the specific growth rate is zero.

3. Between these two extremes, the specific growth rate makes a smooth transition.

This model considers each cell to be equally contact inhibited. However, only cells on the interior of a patch are actually contact inhibited. The cells on the perimeter grow at the normal rate.

Cherry and Papoutsakis presented an improved deterministic model in [11]. This model takes into account different growth rates based on the location of the cell in the patch. The following criteria were used:

1. Cells on the interior of a cell patch have a zero growth rate.

2. Cells on the perimeter of a cell patch grow at the normal rate.

3. Patches of cells are assumed to be circular with particular dimensions based on the shape of the growth surface.

This model overpredicts the growth rate. To compensate for the overprediction, the growth rate changes based on the number of cells per patch. When the number of cells per patch is low, the model uses an exponential growth pattern. Once the cell patch reaches a certain size, the model switches to a zero growth rate.

The deterministic models discussed in this section provide insight into simple cell population dynamics. However, these models overpredict the growth rate, and consider a cell patch to be of a circular shape. They do not compensate for the growth rate in the case of irregular shaped patterns. In addition, neither of these models incorporates cell locomotion nor considers the topological configuration of the cell populations.

2.2 Stochastic Models

To address the issue of cell topology, Lim and Davies developed a stochastic model where the cell shape and cell removal from the colony are taken into consideration [41]. The cell topology is modelled as a matrix of irregular polygons partitioned using the *Voronoi tessellation* technique. Each polygon represents a single cell containing one nucleus. Each

site is either free or occupied by a cell. A cell can only divide if a neighbouring site is free. The model also follows these assumptions:

1. The cells are flattened and the shapes can be represented by external irregular polygons.
2. The positioning of the nuclei of a cell is placed randomly on the substrate at seeding.
3. The maximum cell density at confluence is fixed for the type of cells and the culture conditions.
4. The division time of a perimeter cell is represented by a normal probability distribution.

This model accounts for the formation and the merging of cell colonies. Unlike the model by Cherry and Papoutsakis, it does not assume the shape of the cell colonies to be circular. The removal of cells from the colony by shear forces or death was also considered in the model and found to affect cluster shapes. This two-dimensional model made some restrictive assumptions on cell-cell interactions and did not take into account cell locomotion.

Ruaan, Tsai, and Tsao proposed a stochastic model for the simulation of density-dependent growth of anchorage-dependent cells on flat surfaces [54]. They based their model on experimental observations of Chinese hamster ovary (CHO) cells obtained using time-lapse video microscopy techniques. Their model attempted to incorporate the effects of cell motility and considered that the cell sizes varied with time. Based on their experimental observations, the model of Ruaan et al. assumed that cell migration (or displacement) occurs only during a short period of time after division. Thus, the daughter cells that find themselves in close proximity to each other after a division are displaced and can appear at low-density areas within a circle defined by the average motility and the doubling time. One of the limitations of this model is its assumption that cells move within the first hour after cell division in accordance with the experimental observations on the movement of CHO cells. However, this assumption is not valid for other types of cells. In addition, this model did not take into account the cell-cell interactions and collision phenomena.

2.3 Cellular Automata Models

Zygourakis, Bizios, and Markenscoff developed a two-dimensional model based on cellular automata [64, 65]. This model allows for contact inhibition during the proliferation process. A cellular automaton is defined as a two-dimensional network of computational sites. Each site is a two state automaton that can be either occupied by a living cell or free and available for cell growth.

Using the cellular automata concept, Forestell, Milne, and Behie [16] as well as Hawboldt, Kalgoreakis and Behie [24] modelled cell growth on microcarriers. In the former model, the sites are labelled as occupied and unoccupied. In addition, a site that is set for cell division on the next iteration is marked as newly occupied. There are two key rules in the implementation:

1. A site which was occupied at the beginning of the current time step remains occupied at its end.

2. If an occupied site had one or more unoccupied sites in its neighbourhood, then one of the unoccupied sites was selected at random and marked newly occupied.

In the model proposed by Hawboldt, Kalgoreakis and Behie [24], the surface of a microcarrier was described in the form of a neighbour table, which is a matrix defining a cell and its neighbouring cells. A site may have one of the four states: unoccupied, occupied, newly occupied, and inhibited.

Further, Lee, McIntire and Zygourakis showed the importance of cell motility and cell-cell interaction in describing the cell proliferation rates [35]. Any comprehensive model for tissue growth must consider these processes and account for the growth factors that regulate their rates. Their experimental studies also elucidated the fundamental mechanisms of endothelial cells motility and proliferation. Lee, Kouvroukoglou, McIntire and Zygourakis followed this work with a new model [36] that described the locomotion of migrating endothelial cells in two-dimensions using a set of parameters that included the speed of cell locomotion, the expected duration of cell movement in a given direction, the probability distribution of turn angles which decides the next direction of the cell movement, and the frequency and duration of cell stops.

Building on these early models, Ben Youssef developed the first three-dimensional cellular automata model for tissue growth [3]. This work also utilizes a Markov chain approach to

model the trajectories of migrating cells [4] and is focused primarily on the study of a single population of proliferating and migrating cells. It forms the basis for our model. Cheng, Ben Youssef et al. enhanced this model by accounting for the slowing of cells resulting from cell-cell interactions. The model used a second stationary state to represent the "stickiness" of cells. The "stickier" the cells are, the higher their tendency to form multicellular aggregates.

Chang and co-workers developed a 3-D cellular automata based model to represent the growth of microbial unit cells [9]. This model considers the effects of bacterial cell division and cell death. It does not consider, however, contact inhibition and cell migration. Here, cells can divide into one of 26 directions, corresponding to all the possible adjacent cubes of a dividing mother cell. When a cell divides, space for a daughter cell is made by pushing the entire line of cells between the dividing cell and the closest free space until the adjacent space becomes available. In the event of cell death, the dead cell remains in its former location. As pressure from dividing cells persists, the dead cell eventually breaks down into a free space and is no longer part of the structure formed by the biofilm.

Cellular automata based models have also been used to solve more specific modeling problems. Kansal et al. developed a model to simulate brain tumour growth dynamics [28]. Their model utilizes a few automaton cells to represent thousands of real cells. While this reduces the computational requirements of the model, it also eliminates the ability to control the state of individual cells and prevents their tracking in the cellular space. Another cellular automata based model was used by Cickovski and co-workers in a simulation to model morphogenesis [12]. This model, called COMPUCELL3D, uses a hybrid approach to simulate the growth of an avian limb. COMPUCELL3D uses a cellular automaton to govern the cell interactions while using reaction-diffusion equation solvers to determine the concentration levels of surrounding chemicals. The model simulates the following cellular processes: cell migration, division, death, and cell-cell adhesion, where a cell can span multiple lattice sites. This model is focused on simulating morphogenesis in multicellular and unicellular organisms.

2.4 Chapter Summary

The objective of this chapter was to present and review previous studies and research that were related, at one level or another, to our work. An emphasis on the different types of

models used to simulate tissue growth was put forth. In particular, the class of cellular automata based models was shown to play key roles in the simulation of cell population dynamics, tumour growth, as well as the regeneration of limbs.

Chapter 3

3-D Modeling of Cell Migration and Proliferation

The migration and proliferation of mammalian cells are important processes in biological systems. In this model, we represent a cell as a cubic computational element. This discrete model incorporates the primary features of cell division and locomotion including the complicated dynamic phenomena occurring when cells collide. The model simulates the growth of tissue comprised of multiple cell types where each cell population has its own division and migration characteristics. It also models cell aggregation which takes place during cell-cell interactions.

In the three-dimensional model, each computational site interacts with its neighbours that are to its north, east, west, south (NEWS), and immediately above it or below it as shown in Figure 3.1. This is the von Neumann neighbourhood in three dimensions. In this case, the neighbourhood radius is equal to one. The choice of this type of neighbourhood has the impact of reducing the algorithm/programming complexity and possibly its computational time requirements.

3.1 Modeling of the Biological System

In biological systems, there are many processes operating at the cellular and tissue levels. Our model focuses on those processes involved in cell migration and proliferation. In particular, it focuses on the following four cell activities:

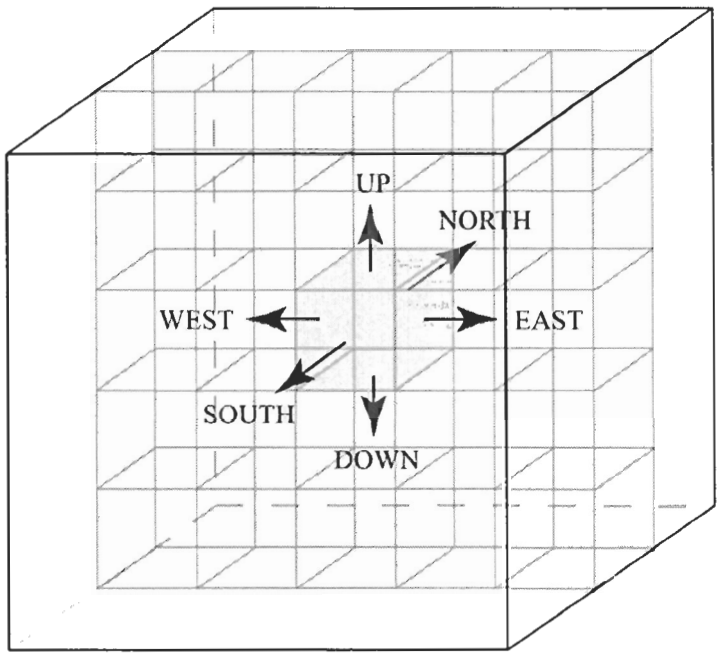


Figure 3.1: The von Neumann neighbourhood in three dimensions.

- Cell division,
- Cell motion,
- Cell collision, and
- Cell aggregation.

3.1.1 Cell Division

Mammalian cells are anchorage-dependent and require a substrate or a scaffold in which they can grow. The growth of cells is characterized by the formation of confluent layers in which neighbouring cells touch one another. As cell division occurs, two newly formed daughter cells are born. Cell division continues until a cell is completely surrounded by other cells. This is known as *contact inhibition*. As confluent patches of cells form, only cells at the outer edges of the patches can divide while cells inside the patches are contact inhibited. The effects of contact inhibition can be reduced with cell motion as increased motility can significantly enhance cell proliferation rates. These two competing processes

and their opposing effects complicate the dynamics of cell population.

We model cell division as a two-step process:

1. If there are vacant sites in a cell's neighbourhood, then the cell divides and two new cells of the same type are created. One daughter cell occupies the original site, and the other daughter cell occupies a vacant neighbouring site.

2. Assign to each of the daughter cells new parameters based on its own cell population characteristics. These are described in a later section.

The position for a daughter cell is chosen according to a random algorithm based on the growth probabilities. The time interval between the division of a cell and its subsequent division is called the cell cycle time. The cell cycle time is known to follow a wide distribution, resulting in the asynchronous proliferation of cells. Once a cell has been contact inhibited, it remains so unless more area becomes available.

3.1.2 Cell Motion

Cell locomotion is an important function of mammalian cells. Cell motility is vital for many physiological processes such as wound healing. It also counterbalances the effects of contact inhibition. Increasing the speed of cells has a positive effect on the proliferation rate of the cell populations, and in the case of wound healing, this aids in the healing process [44]. To model cell motion, we must characterize the trajectories traversed by individual cells.

In earlier studies concerning cell motion trajectories, the common method used was to observe the initial and final cell positions. Such approaches do not provide detailed information on how individual cells move and are incapable of evaluating the important locomotory parameters (such as individual cell speed, persistence, etc.) [3]. Using time-lapse recording techniques, the authors of [65] were able to observe and analyze the migration of bovine pulmonary artery endothelium cells. They noted that these cells moved in a random manner, and implemented a cellular automaton model based on random walk theory to describe the locomotion of these cells. Later studies have shown that these cells execute a persistent random walk [19, 23]. Cells executing such persistent random walks move in a certain direction for a fixed duration of time, and then suddenly turn and move in another direction. Changes in direction can be the result of cell collision or as a response to some intracellular activity. In order to model the characteristics of cell migration, we require at

least the following parameters: turning angles, preferred direction, individual cell speed, and persistence.

Although the persistent random walk model has been successful in assaying and comparing the motility of cells, it does not provide all the locomotory parameters necessary for a detailed characterization of the cell migration process. The authors of [36] have developed a Markov chain model to characterize the locomotion of endothelial cells in two dimensions. This was later extended to three dimensions [3, 4]. Our model uses this approach to describe the movements of individual mammalian cells. To characterize cell migration in a way that is suitable for a computer implementation of a discrete model, the following information is required:

1. The speed of cell migration.
2. The expected duration of cell locomotion in any particular direction.
3. The probability distribution of turning angles to determine the next direction of cell movement.
4. The frequency of cell stops.
5. The duration of cell stops.

3.1.3 Cell Collision

Mammalian cells move in a certain direction for some period of time and, then, they turn and migrate in another direction. Cell collision occurs when a cell moving in a certain direction encouters another cell of a different type in its path. When a cell of one type collides with a cell of a different type, the cell slows down for a period of time and then changes its direction. The duration of time a cell stays in the stationary state after a cell collision is based on its type [37, 32]. Once the cell is ready to resume its motion, it migrates in a new direction. The modeling steps used in the cell collision process are described below:

- The cell stops for a number of steps.
- The cell changes its direction and resumes motion.

In this model, cell collisions can occur either when a cell is moving in a given direction, or when a cell is changing its direction and encounters another cell of a different type.

3.1.4 Cell Aggregation

Cell aggregation is a feature of tissue formation that allows the binding of cells of the same type. It is this specific grouping of cells that enables the tissue to perform its intended purpose. Cell aggregation is the combination of two cellular functions: cell-to-cell recognition and cell adhesion. The self-recognition quality lets cells identify cells of the same type. When cells of the same type encounter each other, they adhere to one another and form a cellular aggregate [61]. As more cells of the same cell type encounter the cellular aggregate, the cellular aggregate becomes larger forming a cluster of cells. The adhesion can be strengthened by aggregation factors that are sometimes secreted by the cells [15]. As a result of aggregation, the cell slows down, "sticks" to another cell of the same type, and changes its direction of motion. The basic steps used to model the process of cell aggregation are listed below:

- The two cells stop for a number of steps, thereby entering an aggregation state.
- The two cells change their direction and resume locomotion.

We limit cell aggregation to cells of the same type. Cell aggregation occurs when a motile cell "collides" with another similar cell.

3.2 Markov Chain Theory

We follow the same Markov chain approach used in [36] to model the trajectories of motile cells. Let C(x, y, z) be the coordinates of a cell C at time t in a three-dimensional *Cartesian* coordinate system. In Figure 3.2, $C'(x, y,0)$ represents the projection of C onto the xy-plane. The same figure also shows that an angle θ is formed by C' and the positive x-axis, while an angle ϕ is formed by C and the positive z-axis.

In three dimensions, the two angles θ and ϕ can be used to completely describe the direction of the motion of a cell. We can further illustrate this in Figure 3.3. Here, we decompose the state space into two subdomains. In Figure 3.3(a), we define a reference axis for the first domain formed by an angle equal to $-(\pi/4)$ with the positive x-axis. Cell C is in state j, $j \in \mathbb{N}$, when the angle θ is in the interval $\theta_{j-1} \leq \theta < \theta_j$. The second subdomain displayed in Figure 3.3(b) has a reference axis perpendicular to the z-axis, and is formed

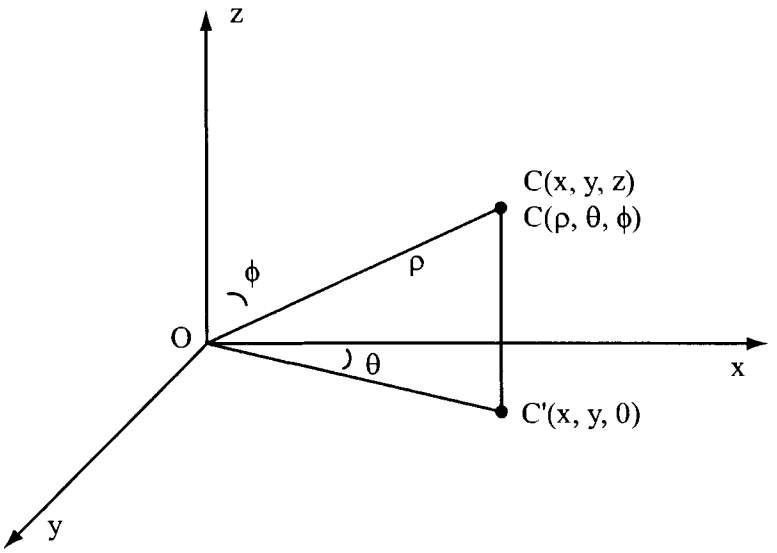


Figure 3.2: The location of a cell, C, in relation to two three-dimensional coordinate systems (the Cartesian and spherical ones).

at an angle of $\pi/2$ with the positive z-axis. Cell C will be in state $(j+4)$ if the angle ϕ lies in the interval $\phi_{j-1} \leq \phi < \phi_j$.

In our Markov chain model, we define the six directional states as combinations of the state spaces defined by the two angles θ and ϕ. This means that the state space of 2π defined by the angle θ is divided into four equal parts to yield the first four directional states. In addition, the other state space of 2π defined by the angle ϕ is then divided into two equal parts to yield the last two directional states. Therefore, cell C is in state 1, if it moves in a direction with angle θ in the range $0 \leq \theta < \pi/2$. The same cell will be in state 2, if θ is in the range $\pi/2 \leq \theta < \pi$, and so on. Applying this definition to angle ϕ, cell C is in state 5 $(= 1+4)$ if ϕ is in the range $0 \leq \phi < \pi$ and will be in state 6 $(= 2+4)$ if ϕ is in the range $\pi \leq \phi < 2\pi$. The above decoupling assumes that there is no simultaneous changes in θ and ϕ. In addition to the above states, a cell is said to be in a stationary state (state 0) if it does not move more than a half of a cell diameter between two consecutive time intervals. This is known as the half-cell diameter rule [47].

Next, we define the parameters used in the Markov chain model to describe the trajectories of mammalian cells. These are as follows:

- Transition-state probabilities: These probabilities characterize the turning behaviour

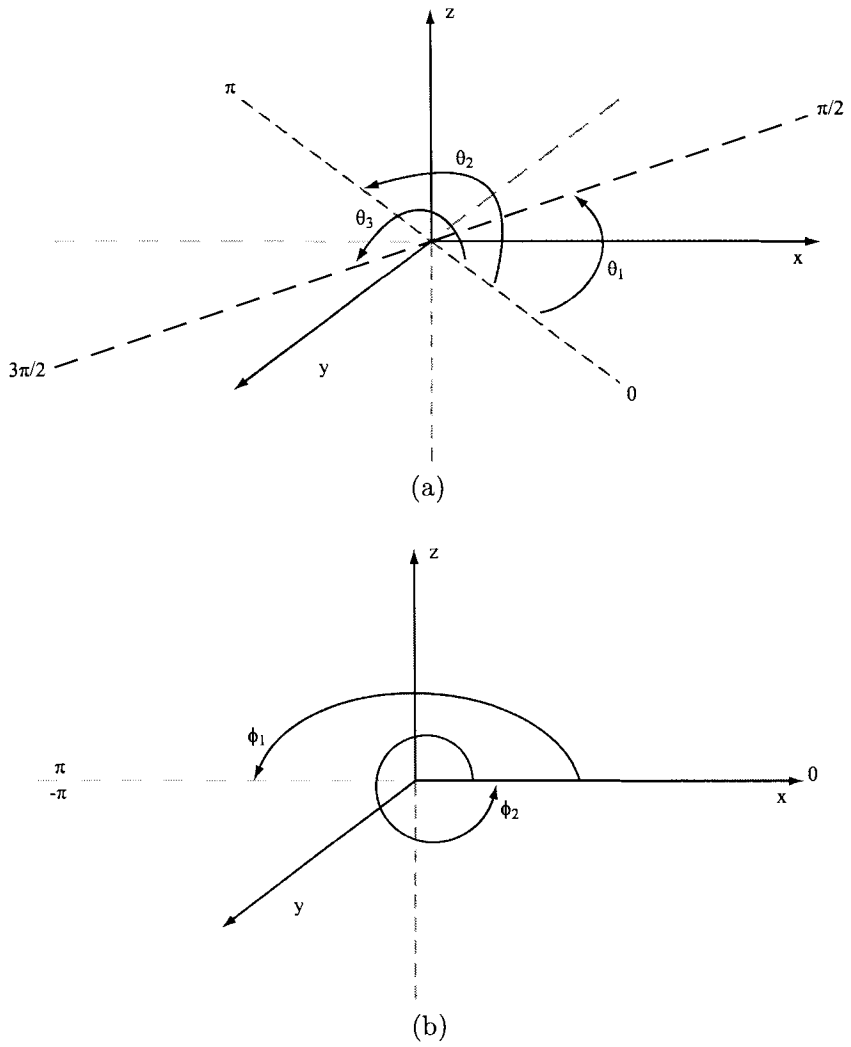


Figure 3.3: Illustration of the states of the discrete-time Markov chain model. The two state spaces of 2π each are divided into (a) one with four equal parts where each section spans $\pi/2$ radians, and (b) another with two equal parts where each section spans π radians. In both parts, the reference axis/plane is labelled with 0.

of cells by quantifying the frequency at which the cells move from one state to another.

- Waiting times: These are the average times cells spend in a particular state.

We let X be the set of all states of a Markov chain, where X contains the six directional states and the two stationary states (state 0 and state 7). We will describe each cell trajectory as a stochastic process $[X_t, t \geq 0]$, where X_t is a random variable at time t that can take any value from the set of possible states X. Let $l \in X$.

Below, we define the *memoryless* property of the Markov chain as given in [4]:
Definition: A stochastic process $[X_t, t \geq 0]$ is a discrete time Markov chain if for all times $0 \leq t_1 < t_2 < \ldots < t_n < t$ and $\tau \geq 0$, the transition probabilities p_{il} are given by

$$p(X_{t+\tau} = l | X_t = i, X_{t_n} = i_n, \ldots, X_{t_1} = i_1) = p(X_{t+\tau} = l | X_t = i),$$

for any states $(i_1, i_2, \ldots, i_n, i,$ and $l)$.

This property states that at time $t + \tau$ the probability of a cell being in state l is independent of how it entered state i at time t. This memoryless property of Markov chains can be summarized as follows: The probability that the process will be in a given state at the next time step is deduced from its state at the present time and does not depend on the history of the process.

Let T_i be a random variable defining the time a cell spends at state i every time it visits that state. For Markov processes, the random variable T_i, called the waiting time in state i, has an exponential distribution with parameter $\lambda(i)$. This exponential probability distribution function can be expressed by

$$p(T_i > t) = \int_t^\infty \lambda(i) e^{-\lambda(i)\tau} d\tau.$$

From the properties of the exponential distribution [29], the expected value of T_i is given by

$$E[T_i] = \int_{-\infty}^{\infty} t\lambda(i) e^{-\lambda(i)t} dt = \frac{1}{\lambda(i)}.$$

Both the expected values of the waiting times and the transition state probabilities are input parameters for our model whose values can be experimentally determined. Another property of Markov chains is the possession of stationary time-invariant transition probabilities

$p_\tau(l|i)$. This probability is illustrated by the following equation:

$$p_\tau(l|i) = p(X_{t+\tau} = l|X_t = i), \forall t \geq 0, \text{ where } \tau \geq 0 \text{ is fixed.}$$

This is a statement that the probability of switching from state i to state l at time $t + \tau$ is not dependent on how long the process was in state i.

3.2.1 Computation of Cell Locomotion Parameters

The speed of migration can be calculated using the cell trajectory data. If $d_i, i = 1, 2, \ldots, N$, are lengths of the cell trajectory segments travelled between time t_{i-1} and t_i ($t_i = t_{i-1} + \Delta t$), then the cell speed of migration in the direction of displacement d_i over the time increment Δt can be calculated by:

$$v_i = \frac{d_i}{\Delta t}.$$

The average speed over the time interval $[0, N(\Delta t)]$ is given by:

$$\bar{v} = \frac{\sum_{i=1}^{N} d_i}{N(\Delta t)}.$$

The population-average speed of locomotion can be computed by averaging the speed of all cells. We can simplify the calculations by defining the size of a computational site, h, as the average size of a living cell. For our simulations, we base the size of the computational site on the average area of mammailan cells at confluence. This sets the length of a side for each computational site to be $10\mu m$. If $\Delta t = 0.2hr$, then the instantaneous speed of a migrating cell moving in any one of the six specified directions is $50\mu m/hr$. By adjusting the value of Δt, a new instantaneous speed of locomotion can be calculated. These calculations can be used to monitor the temporal evolution of the average speed of the individual cell populations. After every iteration, the sum of distances covered by all migrating cells is calculated. At that instant, the population-average speed of locomotion is computed by dividing the sum by the total number of cells, migrating or not, by the time interval Δt.

3.3 New Features of the Model

We enhanced the existing model by adding new features that allow for the modeling of the proliferation and migration of multiple cell types. These features are described below.

3.3.1 Cell Aggregation

In addition to implementing cell collision, we modelled cell aggregation by adding a second stationary state (j = 7). Two colliding cells of the same cell type will enter the aggregation state and stick together for a defined period $E(T_7)$. The value of $E(T_7)$ indicates the likelihood for the cells to form multicellular aggregates. When a cell collides with another cell of the same type already in the aggregation state, then the first cell will enter the aggregation state and both cells will reset their persistence counters to $E(T_7)/\Delta t$. Once the waiting time $E(T_7)$ has expired, the two cells can move away in randomly assigned directions.

3.3.2 Cell Topologies

This thesis considers two types of initial seeding topologies: uniform topology and "wound" seeding topology. Each topology was chosen for its applicability to research conducted in laboratories. In the uniform topology cells are randomly seeded in the cellular space. This topology simulates the migration and proliferation of cells in a sparsely and uniformly seeded environment with the objective of simulating tissue regeneration. In the "wound" seeding topology, an empty cylinder at the centre of the cellular space is surrounded by seeded cells in the remainder of the space. Cells bordering the empty core migrate into the centre and proliferate. With each of the above two cell seeding topologies, we have associated two types of cell distributions as described below.

Segmented Distribution

The first cell distribution is the *segmented distribution.* Here, each cell type is seeded in a separate area of the cellular space. During the simulation, cells can migrate freely in the cellular space, and can enter areas that were originally designated for a particular type of cell during the initial cell seeding. Figures 3.4 and 3.5 show an example of a segmented

distribution in the case of uniform and "wound" seeding topologies of two cell types, respectively. In the uniform seeding topology, the seeding density indicates the percentage of sites occupied by cells at the start of the simulation.

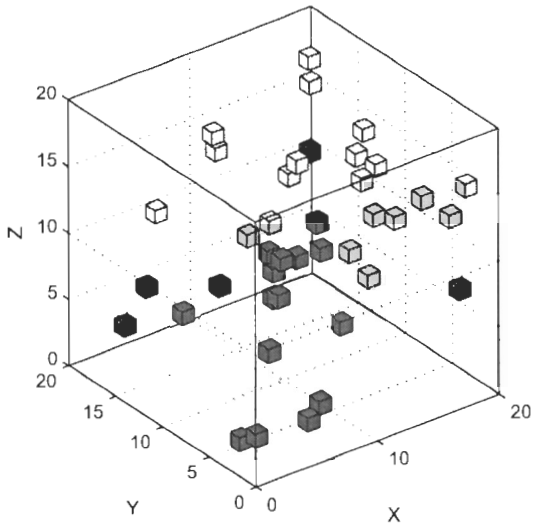


Figure 3.4: An example of a uniform topology using two cell populations in a segmented distribution with a total initial seeding density of 0.5%. This example is based on a $20 \times 20 \times 20$ cellular array.

Mixed Distribution

The other seeding distribution associated with the cell seeding topologies is the *mixed distribution*. Here, the different cell types are seeded together in a random order according to the cell topology. Figure 3.6 shows the mixed distribution of two cell types in a uniform seeding topology, while figure 3.7 shows two types of cells in a mixed distribution in the case of a "wound" seeding topology.

3.4 A Cellular Automaton

Our cellular automaton model performs the simulations on a three-dimensional array of computational sites. A computational site can contain a single mammalian cell. Each cell

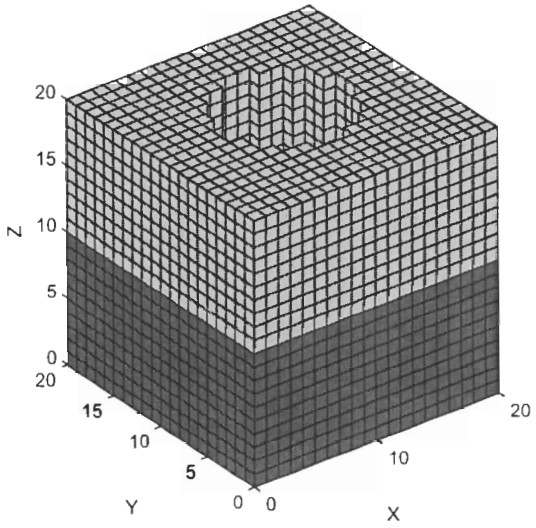


Figure 3.5: An example of a "wound" seeding topology with two cell populations in a segmented distribution. This example is based on a $20 \times 20 \times 20$ cellular array. For the wound, a diameter of 10 and a height of 20 are used.

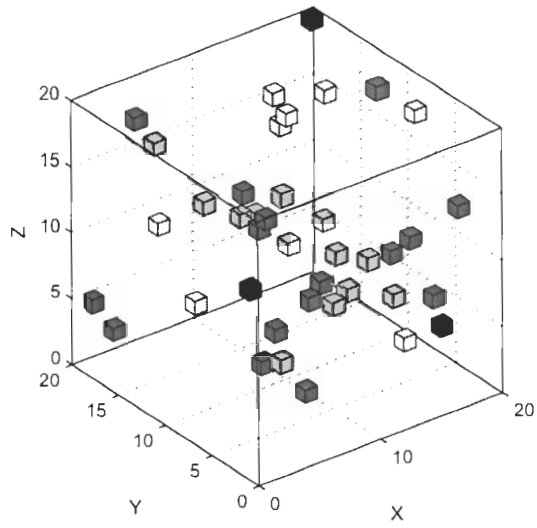


Figure 3.6: An example of a uniform topology using two cell populations in a mixed distribution with a total initial seeding density of 0.5%. This example is based on a $20 \times 20 \times 20$ cellular array.

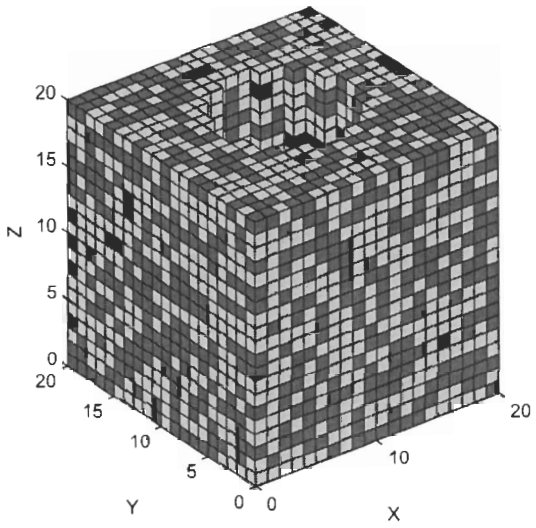


Figure 3.7: An example of a "wound" seeding topology with two cell populations in a mixed distribution. This example is based on a $20 \times 20 \times 20$ cellular array. For the wound, a diameter of 10 and a height of 20 are used.

can be in one of a finite number of states and can interact with a finite number of neighbours. At any instant in time, the computational state of the site describes the characteristics of the cell at a particular time. At discrete times, the computational sites change states by interacting with neighbouring cells.

In modeling the proliferation and migration of mammalian cells, each site can be either occupied by a living cell, or be free and available for cell movement and division. The state of each computational site delineates the properties of a cell, while the collective states of all sites denote the state of the cellular automaton.

3.4.1 States of the Cellular Automaton

Our model is a discrete system operating in a cellular space that is comprised of $N = N_x \times N_y \times N_z$ computational sites. The cells in the cellular space interact with their neighbours at equally spaced time intervals $t^1, t^2, \ldots, t^r, t^{r+1}, \ldots$ (where $t^{r+1} = t^r + \Delta t$ for all r).

The cellular automaton is a dynamic system evolving with discrete parallel iterations.

Each occupied computational site must describe the current state of a given cell. In this model, we define a set of values to represent the state of a cell. These values must describe asynchronous proliferation and persistent random walks of multiple cell types. In building an adequate state definition, sufficient information must be provided about the history so that given the current state, the past is statistically irrelevant for predicting all future behaviour pertinent to the application at hand [7]. This is consistent with the definition of state in a Markov-chain setting. According to these specifications, the state x_i of an automaton containing a living cell must specify the following set of parameters:

1. The type of cell.
2. The direction of cell motion.
3. The speed of the cell.
4. The time remaining until the next direction change.
5. The time remaining until the next cell division.

The average speed of migrating cells is controlled by varying either the value of the time interval, Δt, or the speed factor. This is due to the fact that migrating cells cover a fixed distance in each step. Another means of regulating the speed of locomotion is the ability to adjust the transition probability for the stationary state. Therefore, a migrating cell of type j in automaton i must only specify the direction of locomotion and the times remaining until the next direction change and the next cell division in its state x_i. The state x_i of an arbitrary automaton i, then, takes values from the following set of eight digit integer numbers:

$$\Upsilon = \{klmnpqrs | k, l, m, n, p, q, r, \text{and } s \in \mathbb{N}\},$$

where k is the cell type. The direction of motion is identified by the *direction index* l (see Table 3.1). When l is equal to 0, the cell is in the collision stationary state. When the value of l ranges from 1 - 6, it represents one of six directions the cell is currently moving in. When the value of l is 7, it enters an aggregation stationary state where it "sticks" to another cell of the same type potentially forming cellular aggregates. The digits mn denote the *persistence counter*. It is the time remaining until the next change in the direction of

cell movement. The *cell phase counter* is represented by the digits $pqrs$. The cell phase counter is the time remaining before the cell divides.

Table 3.1: The direction index for migrating cells and the corresponding direction of motion.

Direction Index (k)	Direction of Motion
0	Stationary (Collision)
1	East
2	North
3	West
4	South
5	Up
6	Down
7	Stationary (Aggregation)

The initial value of the cell phase counter $pqrs$ is randomly assigned to each new cell based on the cell division time distribution for that cell's population type. The division time distribution for each population is obtained from available experimental data [35]. During each iteration in the simulation, the phase counter for each cell is decremented by one. Once, the phase counter reaches zero the cell divides. The persistence counter mn is assigned after each direction change. It is assigned the value of the average waiting time based on the application of Markov chain analysis to the cell trajectory data. The persistence counter for each cell is decremented at each iteration, and the cell changes direction when the persistence counter reaches zero.

3.5 Mixed Cell Cultures and Tissue Architecture

Most tissues consist of several types of cells that organize themselves in very specific spatial patterns [48]. This three-dimensional architecture is what endows tissues with the special functions of organs. While previous studies have focused on single types of cells, our model allows for simulating multiple cell types, with each type having its own migratory and proliferation characteristics.

The model allows for the organization of cell populations into specific spatial patterns. Tissue engineers are attempting to achieve this goal by using microlithographic techniques

to create surfaces with heterogeneous characteristics and solids with specific pore structure. When micro-patterned surfaces are used, different cells will migrate at different rates on different parts of the surface. Thus, we can guide cells of certain type to cluster in specific areas only. These areas may be surrounded by cells of another type.

3.6 Chapter Summary

In this chapter, we described the modeling steps of the biological system beginning with the processes of mammalian cell proliferation, migration, collision, and aggregation. We also reviewed the Markov chain approach used to model the trajectories of locomoting cells. We then discussed the new features of the model and presented the states of the cellular automaton to be employed in the model.

Chapter 4

Sequential Algorithm

This chapter discusses the sequential algorithm used to implement the three-dimensional model. We present the main parameters used to run the model and explain the related algorithmic details.

4.1 Sequential Algorithm for Cell Proliferation and Migration

The algorithm is based on an earlier version that deals with a single cell population developed by Ben Youssef [3]. The simulation begins by populating the cellular space with cells based on the cell distribution, seeding topology and density. The states for each cell are randomly assigned based on the population characteristics defined for that cell type. Each cell is assigned values for its type, the direction of motion, the cell persistence and cell division counters based on experimental data. In the following subsections, we describe the input parameters used by the algorithm to characterize the cellular space and simulate the growth of tissue.

4.1.1 Input Parameters

Below, we provide a description of the input parameters used to start the simulation. Note that the values of these parameters are chosen either according to experimental data obtained from previous models [3, 4, 36], or are specified by the user.

Global Parameters

The global parameters define the simulation details of the cellular space and the different types of cell populations.

Computational Array. Three parameters denoted by N_x, N_y, and N_z specify the number of computational sites in each of the three dimensions.

Surface. This is the type of surfaced used (fixed boundaries, wraparound in one, two or three dimensions).

Cell Population Parameters. The following cell parameters define the characteristics for each cell population to be seeded into the cellular array.

Cell Type. This is a unique identifier for each cell population.

Speed Factor. This parameter indicates the number of time steps between movements. It controls the cell speed.

Nwait. This is the average duration of time a cell spends migrating in a direction of motion indicated as a number of time steps.

Nwait0. This is the average duration of time a cell spends in the stationary state indicated as a number of time steps.

Nwait7. This is the average duration of time a cell spends in the aggregation stationary state indicated as a number of time steps.

Hist. This is a set of values indicating the percentage of cells in a population that divide in a specified period of time.

Awg(1:6). This is the cell division probability into each of the six neighbouring sites, respectively.

Seeding Parameters

The seeding parameters define the initial cell seeding distributions in a subsection of the cellular space.

Seed Topology. This is the topology of the seeded cells.

Seed Area and Location. These define the size and location of the localized seeding area.

Overall Seeding Density. This is the total number of seed cells as a percentage of the total number of computational sites in the local seeding area.

Seeded Population. This denotes the type of cells to be seeded into the local seeding area.

Cell Type Seeding Density. This defines the percentage of the seeding density of a given cell type in the local seeding block.

4.2 Algorithm

4.2.1 Initial Condition:

1. Read in the initial parameters from the input data file.
2. Select the computational sites to be occupied by the cells at the start of the simulation based on the seeding shape, the seeding pattern, and the population density.
3. For each occupied site, we assign a cell state based on the population characteristics of that cell type. The direction index is randomly selected, the persistence counter is assigned a properly chosen value, and the cell phase counter is set based on experimentally determined cell division data.

4.2.2 Iterative Operations:

At each time step $t^{r+1} = t^r + \Delta t$, $r = 0, 1, 2 \ldots$

1. Randomly select a computational site.
2. If this site is occupied by a cell, c, and the phase counter is zero then it is time for this cell to divide and the division routine is called (see Section 4.2.3).
3. If this site is occupied by a cell, c, and the persistence counter is zero, then it is time for this cell to change directions and the direction change routine is called (see Section 4.2.4).

4. Otherwise, the cell, c, is occupied and both the phase and persistence counters are not zero. Attempt to move this cell to a neighbouring site in the direction indicated by the direction index of its current state.

   - If c is eligible to move during this time step (the motion step is true), attempt to move c into a neighbouring computational site.
     - If this neighbouring site is free, then mark the site to contain c at the next step and decrement the phase and persistence counters by one.
     - If this neighbouring site is occupied by a cell from a different cell type, then the cell remains in the current site and enters the stationary state. The persistence counter is set to $nwait0$, and their phase counters are decremented by one.
     - If this neighbouring site is occupied by a cell from the same cell type, then the cell remains in the current site and both cells enter the aggregation stationary state. The persistence counters for both cells are set to $nwait7$, and their phase counters are decremented by one.

5. Select another site and repeat steps 2–4 until all sites have been examined.

6. Update the states of all sites so that the locations of all cells are set for the next time step.

4.2.3 Division Routine

1. Determine if there are free adjacent sites surrounding the dividing cell. If all of the adjacent cells are occupied, then the cell is contact inhibited and will not divide. The phase counter is assigned a new value.

2. If there are free sites in the neighbourhood, then select one of these sites by using a random algorithm based on the growth probabilities.

3. Mark the chosen site to prevent other cells from occupying it. This is the site of one of the daughter cells. The other daughter cell will occupy the current location. In the next iteration, assign to the new cell the same cell type index as the parent, a randomly chosen direction index, and new persistence and phase counter values.

4.2.4 Direction Change Routine

1. Determine if there are free adjacent sites surrounding the migrating cell. If all of the adjacent cells are occupied, then the cell remains in its current location, and it is assigned a new persistence counter.

2. If there are free adjacent sites in the neighbourhood, randomly select one of the unoccupied sites based on the state-transition probabilities $p(i|j)$, $i, j = 0, 1, \ldots, 7$.

3. Mark the selected site that will contain the cell in the next time step to prevent other cells from occupying it. Set the persistence counter to its appropriate initial value, and decrement the cell phase counter by one.

4.3 Flowcharts

The following figures contain the flowcharts of the sequential algorithm. The cellular space is assumed to have a fixed boundary.

4.3.1 Main Module

The main module is described in Figures 4.1, 4.2, 4.3, 4.4 and 4.5.

4.3.2 Division Routine

The division routine is depicted in Figure 4.6.

4.3.3 Direction Change Routine

The direction change routine is illustrated in Figure 4.7.

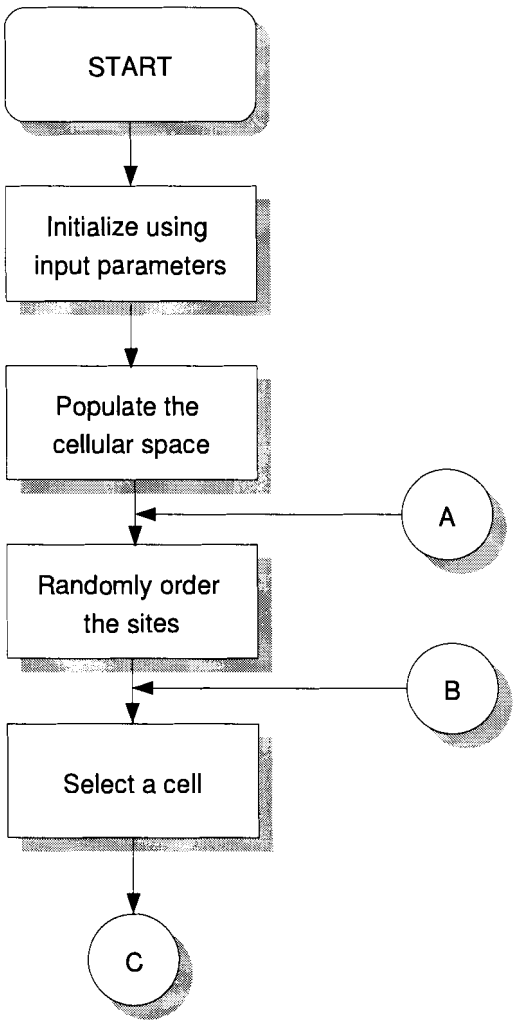


Figure 4.1: Flowchart illustrating the main module of the sequential algorithm (part 1 of 5).

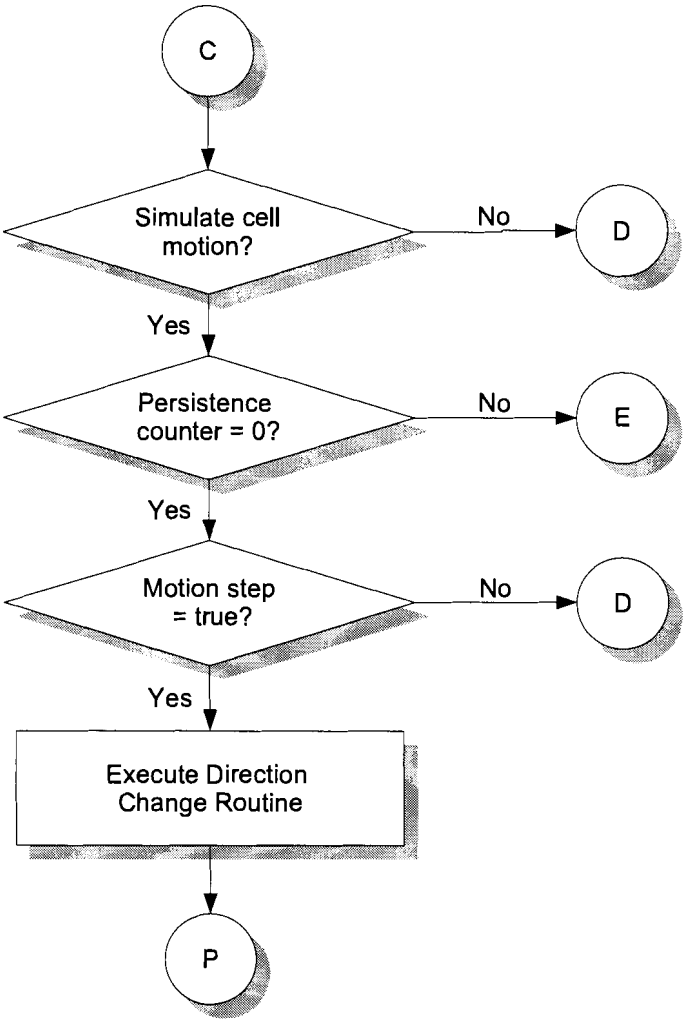


Figure 4.2: Flowchart illustrating the main module of the sequential algorithm (part 2 of 5).

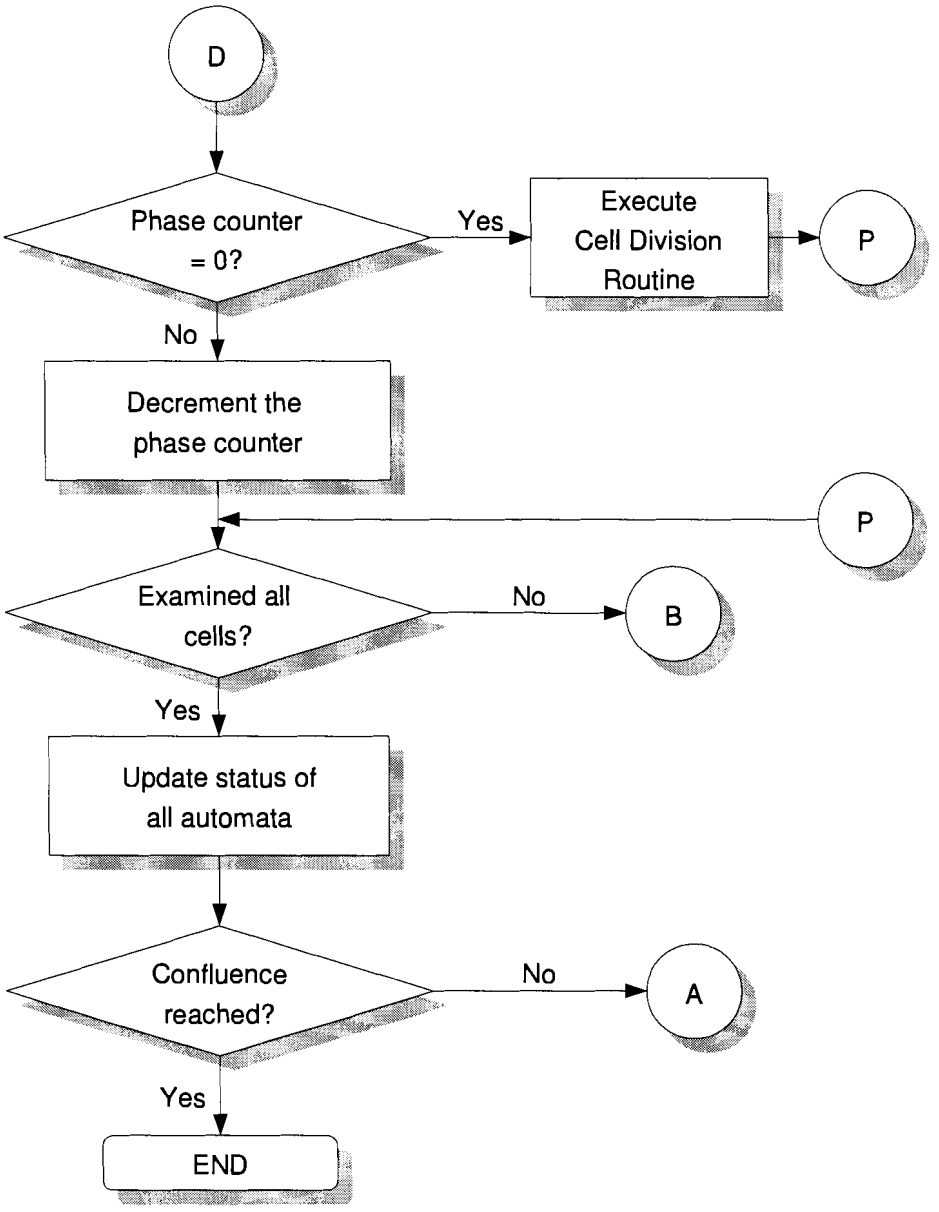


Figure 4.3: Flowchart illustrating the main module of the sequential algorithm (part 3 of 5).

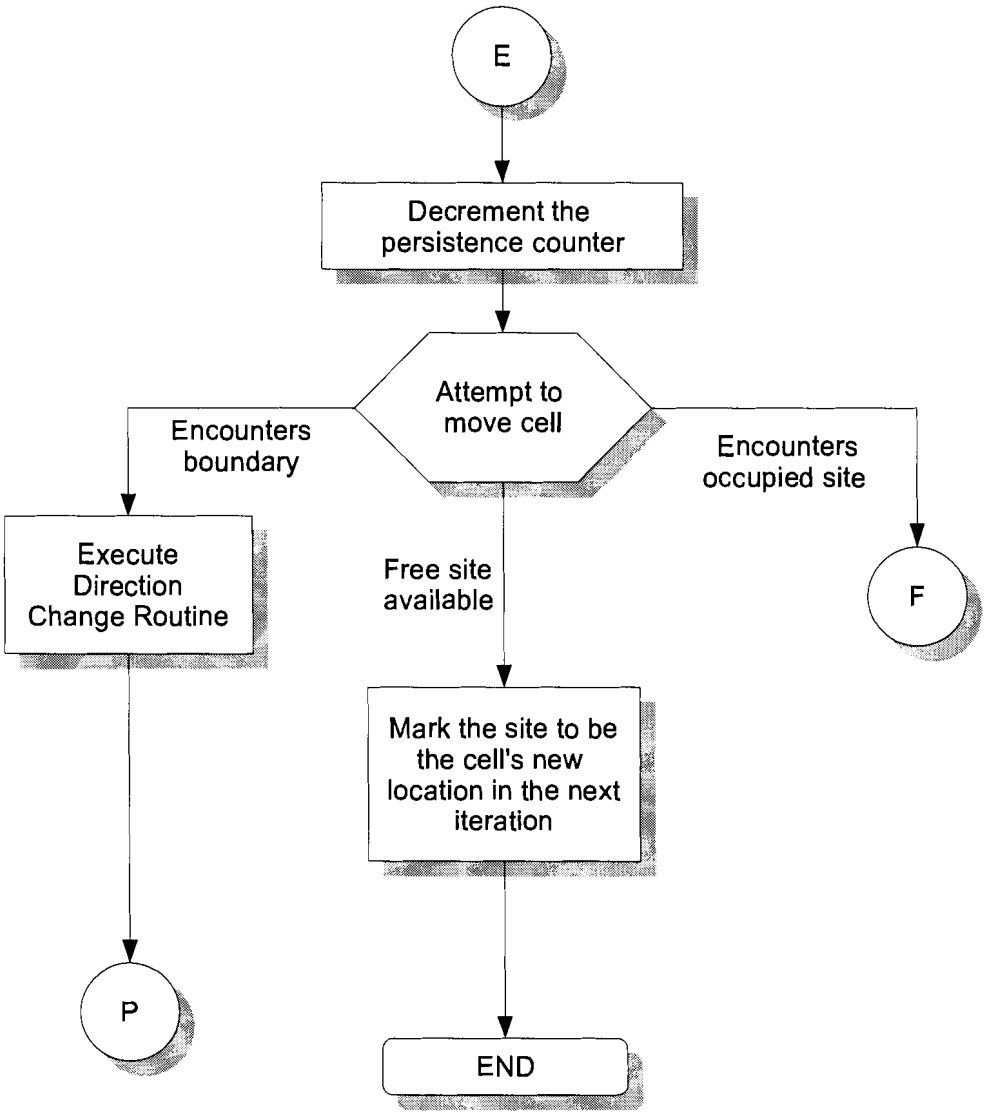


Figure 4.4: Flowchart illustrating the main module of the sequential algorithm (part 4 of 5).

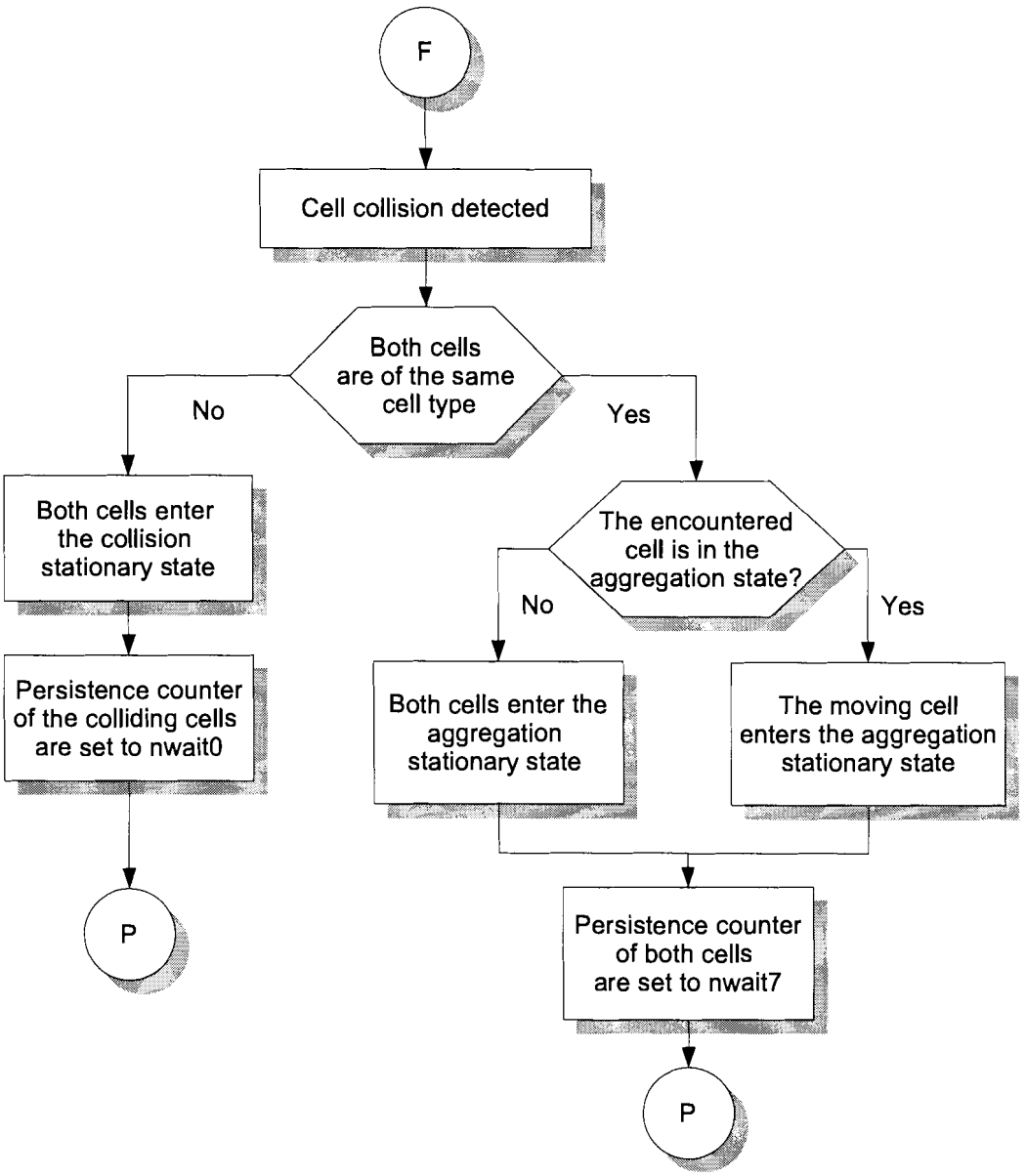


Figure 4.5: Flowchart illustrating the main module of the sequential algorithm (part 5 of 5).

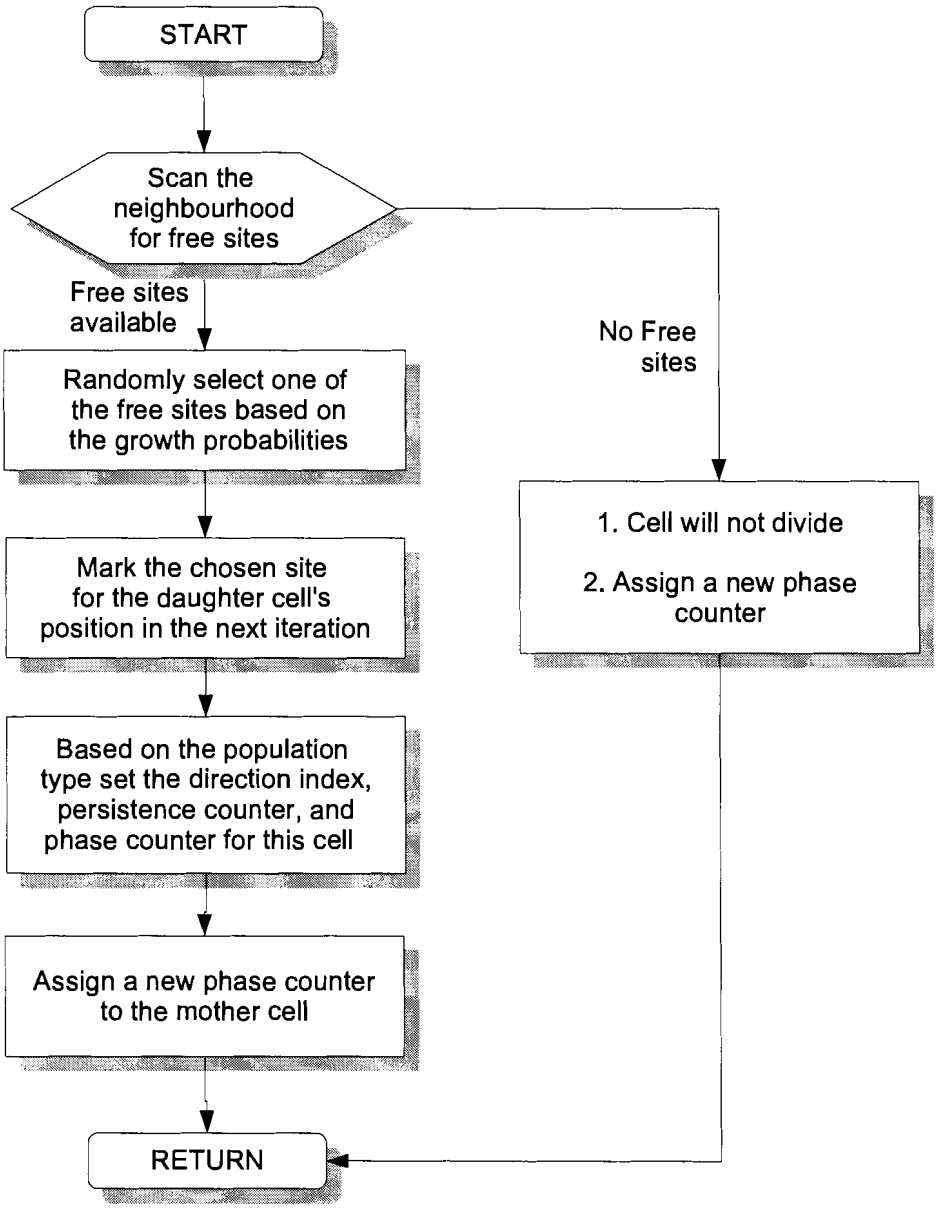


Figure 4.6: Flowchart illustrating the Division Routine.

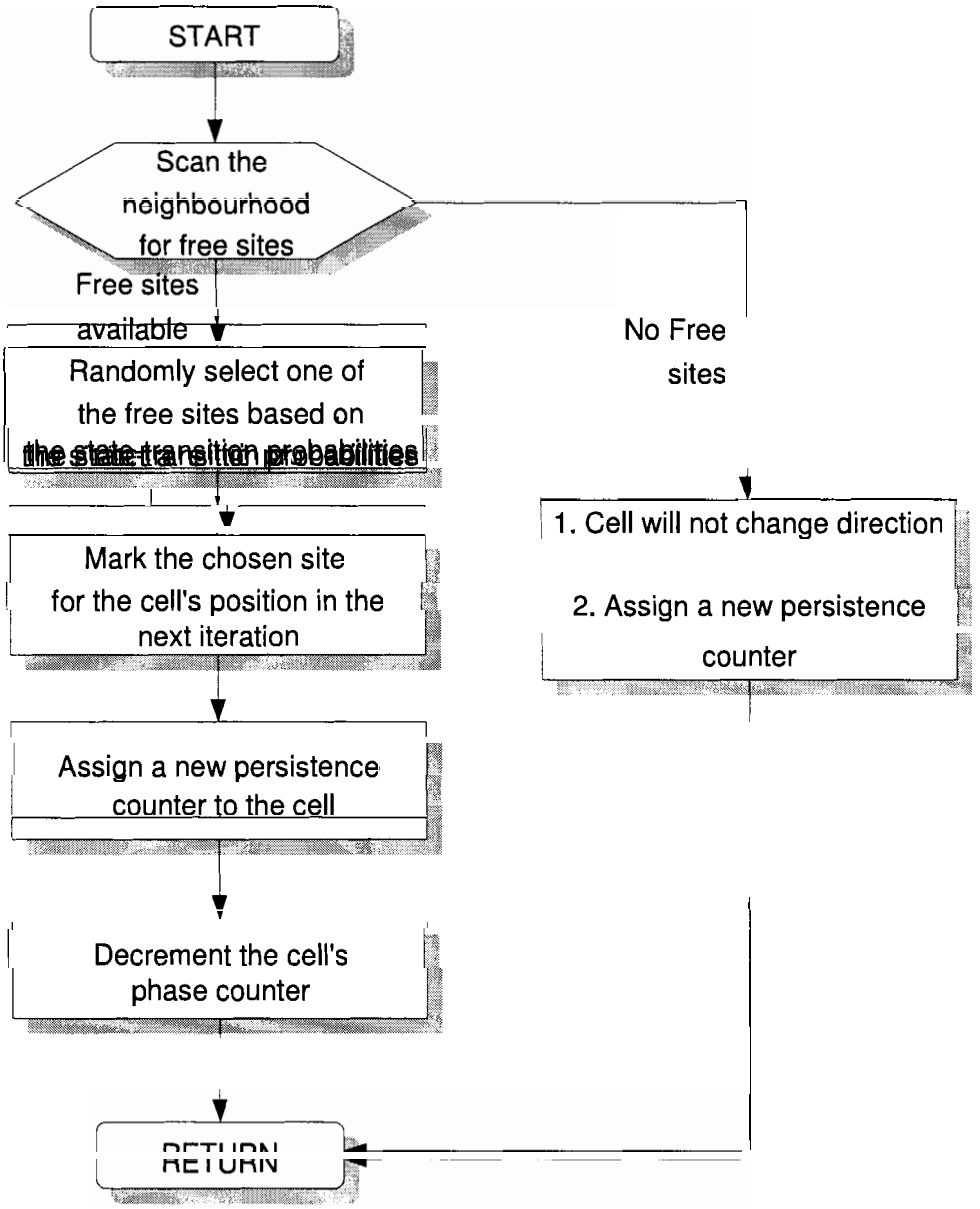


Figure 4.7: Flowchart illustrating the Direction Change Routine

4.4 Other Implementation Details

At every iteration, our model updates the states of all the computational sites. If these sites are scanned in a fixed sequential order, then the processing of the cellular array will be biased based on the fixed ordering. It would be possible, for example, for cells in certain computational sites to be prioritized over other computational sites. This can lead to cells being forced to move in certain directions based on the fixed order in which the cells are being scanned. To eliminate such artifacts, the computational sites must be scanned in a different random ordering at each iteration.

In this model, a random ordering of the computational sites is created at the beginning of each iteration. To create a direct random ordering for a $N \times N \times N$ cellular array would require N^3 calls to the random number generator. This represents computational overhead that can be reduced by using an alternate ordering scheme. We can simplify the computational requirements by creating random orderings for only the N rows, the N columns and the N indices in the third dimension of the array at each iteration. The sites are then scanned by rows according to the random ordering. Within each row, scanning proceeds based on the computed random ordering of the columns. And within each row-column pair, we scan the sites using third dimension indices based on the random ordering of this index. At the beginning of each new iteration, a new random ordering of computational sites is computed. This alternate approach to site ordering requires only $3N$ calls to the random number generator. We expect that there would be no differences in the simulation results obtained by using either of the two random ordering approaches, also known as *complete and partial random orderings*, respectively.

4.5 Random Number Generation

The selection of random numbers is a significant aspect of our model. The model relies on a sequence of uniformly distributed random numbers to determine, for instance, the ordering of computational sites, cell motion, and cell direction. We decided to implement an established algorithm for random number generation rather than using the one provided by the *C++* libraries.

4.5.1 Rationale

The choice to implement a random number generator is based on the fact that random number generators provided by most programming language libraries are not very good [50]. Most developers assume that the random number generator they use is sufficiently random, and thus do not question the validity or the underlying algorithm behind it. In addition, the random number generator can vary based on the compiler used and the operating system the application is running on. By implementing a carefully chosen random number generator, the user is afforded control over the implementation, randomness quality, and periodicity of the generator. This choice also enhances the debugging of the code and facilitates its parallelization [3].

4.5.2 Method Used

Our algorithm uses a random number generator based on the method proposed by D. H. Lehmer known as the *multiplicative linear congruential* generator [38]. This technique is simple and effective in producing uniformly distributed values between 0 and 1. It begins with the careful selection of two fixed integer parameters:

1. A modulus: m, a large prime integer,
2. A multiplier: a, an integer ranging from $2, 3, \ldots, m-1$,

and the subsequent generation of the integer sequence $x_1, x_2, x_3 \cdots$ via the iterative equation,

$$x_{n+1} = f(x_n), \text{ for } n = 1, 2, \ldots,$$

where the *generating function* $f()$ is defined for all x in $1, 2, \ldots, m-1$ as

$$f(x) = ax \bmod m.$$

The sequence x_n must be initialized by correctly choosing the initial integer value x_1. This value, known as the *seed*, is selected from $1, 2, \ldots, m-1$. The sequence is then normalized to the unit interval via division by the modulus operator to produce the real sequence $u_1, u_2, u_3, \ldots$, where

$$u_n = x_n/m, \text{ for } n = 1, 2, \ldots.$$

The choice of m as a prime number ensures that the values of u would vary from $\frac{1}{m}$ to $1 - \frac{1}{m}$ and more importantly prevents the sequence from collapsing to zero. Also, the normalization step does not affect how random the sequence appears. By observing the sequence of x_n, we can determine the overall quality of the randomness. As a result, by properly selecting the multiplier and prime modulus, the resulting sequence can be statistically indistinguishable from a sequence drawn at random (without replacement) from the set $1, 2, \ldots, m-1$ [50]. In our model, we use the Mersenne prime number $m = 2^{31} - 1 = 2,147,483,647$ as a modulus, and the multiplier is set to a = 62,089,991.

4.6 Chapter Summary

In this chapter, the sequential algorithm describing the three-dimensional model was discussed and its representative flowcharts were later given. We also elaborated on other aspects of the sequential algorithm related to the implementation of the random processing of computational sites in the cellular array and the choice of a random number generator.

Chapter 5

Parallel Algorithm

This chapter discusses the parallel implementation of the model on a distributed-memory parallel machine. In particular, the steps necessary to parallelize the sequential algorithm using the Message Passing Interface (MPI) are described. This is followed by presenting the flowchart for a single node program as well as the computing platform.

5.1 Motivation for Parallelization

One of the main objectives of using parallelism is to reduce the overall time required to carry out and complete given computations. This is usually referred to as the *wall clock time* since we are not really interested in some internal measure of performance but rather in the time that we have to wait to obtain the results [3]. Parallelism overcomes some of the constraints imposed by uniprocessors. Besides offering faster solutions, applications that have been parallelized can solve bigger, more complex problems whose input data or intermediate results exceed the memory capacity of a single-CPU computer. Simulations can be run at finer resolutions. Physical phenomena can be modelled more realistically.

5.2 Parallelization and Optimization of the Model

A parallel program is produced by applying the following three steps ([6]):

Decomposition. The division of the program into a set of parallel processes and data.

Mapping. The way processes and data are distributed among nodes.

Tuning. Alteration of the application to improve performance.

Inherent in this process are certain principles that are guides to designing an efficient program. These principles help us determine each of the steps specified above:

- Balancing the computational load.
- Minimizing the communication to computation ratio.
- Reducing sequential bottlenecks.
- Making the program *scalable* (independent of the actual number of available nodes).

Our application is *loosely synchronous* [49]. The amount of computation can vary from one partition and time step to the other as it depends on the amount of useful data. Parallelism is introduced by dividing the work among multiple nodes at each time step. When all nodes finish their parts of the problem, they exchange intermediate results before proceeding to the next time step. Each node has to make sure that the other nodes are ready for exchange of data so that useful data are not overwritten. Between these points the nodes proceed at their own rates. Since the workloads vary temporally and spatially, they have to be distributed evenly amongst the nodes.

5.2.1 Domain Decomposition

We employ a *slab* decomposition technique to divide the cellular space along the z dimension into smaller and more manageable subdomains. The resulting "slabs" are logically mapped to a linear array of nodes. Figure 5.1 illustrates the resulting mapping of the cellular array onto $P = K$ processors. If K divides N_z, then the subdomains will be of equal dimensions, where each subdomain is $N_x \times N_y \times \frac{N_z}{K}$. When $N_z = c_z K + r_z$, the array is divided into r_z subdomains of size $N_x \times N_y \times (\lfloor \frac{N_z}{K} \rfloor + 1)$ and $K - r_z$ subdomains of size $N_x \times N_y \times \lfloor \frac{N_z}{K} \rfloor$. The area of the boundary between two neighbouring subdomains is equal to $N_x \times N_y$ sites. Except for the subdomains at both ends of the decomposition, all the remaining subdomains have two neighbours. In this decomposition, varying the number of processors P (or, subdomains K), does not change the size of the boundary area between two adjacent subdomains. We use the slab decomposition for both the uniform and "wound" seeding topologies.

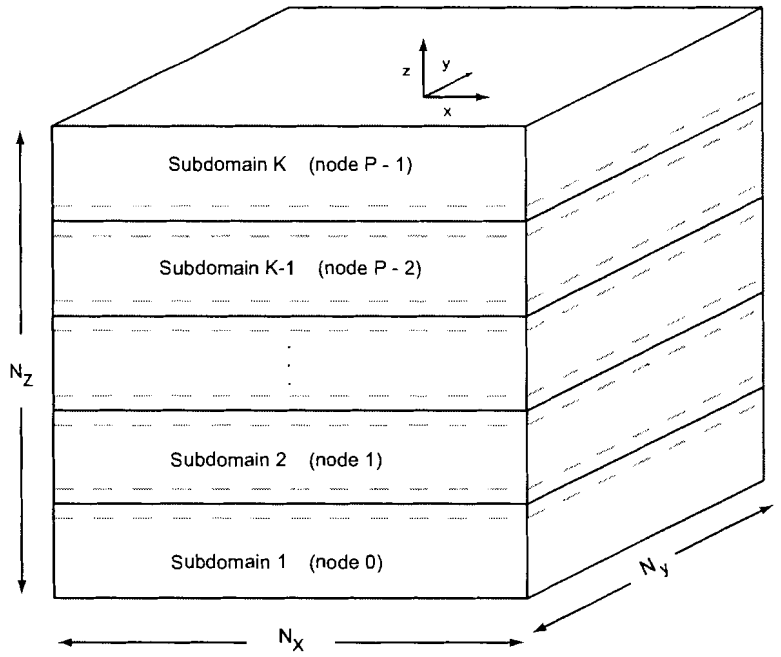


Figure 5.1: Slab decomposition of a $N_x \times N_y \times N_z$ cellular array on $P = K$ processors. The dotted lines denote shared boundaries with neighbouring processors. The node IDs vary from 0 to $P - 1$.

5.2.2 Mapping

While the decomposition of the problem determines whether it is possible to balance the load for deterministic algorithms, mapping is the step that directly determines the load balance. Ideally, all nodes dedicated to an application should be busy during the entire time the application is running. When the work load is not equally distributed among the nodes, nodes with smaller loads would sit idle while nodes with the larger workloads continue to process data.

In addition, there are managerial tasks that are required to be completed. In our algorithm, each node needs to calculate the local volume coverage, cell locomotion speed, and growth rates. An efficient way to collect these data and prepare them for output is to assign a single node (in this case, *node* 0 called the *master*) to preform these functions. This maintains a fairly well-balanced load.

Our algorithm is scalable, in that it can be run on any number of nodes. It is generally a poor approach to create parallel applications for specific number of nodes. The availability

of nodes on a cluster can often dictate how many nodes an application can use at a given time. This availability can vary due to the number of users, removal of nodes, or even node failures. Making the application independent of the current number of available nodes makes the software less dependent on the exact configuration of the hardware.

MPI provides a collection of routines to aid in the development of scalable applications. In particular, there are routines that determine the number of nodes in the system and one that uniquely identifies the current node. These are called `MPI_COMM_SIZE()` and `MPI_COMM_RANK()`, respectively. Using this information, a logical topology can be created based on this naming scheme. In the slab decomposition, we order the nodes from *node* 0 to *node* $P-1$, where P is the number of available processors. While we can establish different logical processor topologies using MPI, this does not reflect the exact topology of the underlying physical hardware. The virtual topology can be exploited by the runtime system in the assignment of processes to physical processors, if this helps to improve the communication performance on a given machine. In our application, the linear array topology reflects the logical communication pattern of the processes.

5.2.3 Tuning

We know that sequential processing is a bottleneck. While some sequential processing is required, some of it can be avoided. A way of reducing some of the serial processing is by utilizing global reduction operations to compute needed simulation parameters requiring data from all nodes. For example, the global summing operation iteratively pairs nodes to exchange their current partial sums. Each partial sum received from another node is added to the sum at the receiving node. Next, this new sum is then sent out in another round of message exchange. The master node, *node 0*, then performs the final operation, be it an addition or division.

In global reduction routines, not every node communicates directly with every other node. Accumulated totals are received at each node, processed and the new accumulated value is distributed in the next message passing round. Only $\log_2(P)$, where P is the number of nodes, rounds of message exchanges are required to complete the operation. By handling the communications in parallel, between pairs of nodes, fewer rounds of message passing are needed. This parallelism reduces the effective number of messages, thus enhancing the performance. Using the global summing routine to count the number of cells in the cellular space, for instance, makes the message passing more efficient, more readable, and minimizes

the communication to computation ratio.

MPI defines a set of collective communication operations known as *reduction operations.* In a global reduction operation, all the processes contribute data that are combined using a binary operation. Some of these binary operations include sum, max, and logical AND, etc. The `MPI_REDUCE()` function performs such global reduction operations in MPI. The `MPI_Op` argument in the `MPI_REDUCE()` function call allows the user to specify the type of operation to be performed. Some examples of `MPI_Op` operators are `MPI_SUM`, `MPI_MAX`, and `MPI_LAND`. When multiple processors require the final result, the `MPI_ALLREDUCE()` function can be used to return the final value to all nodes.

5.3 Flowchart of a Node Program

Figures 5.2 through 5.11 depict the flowchart of the parallel algorithm. The same copy of the parallel program runs on every node of the cluster, as per the SPMD (Single Program Multiple Data) programming model. We also assume the cellular space to have a fixed boundary.

5.3.1 Main Module

The main module is shown in Figures 5.2, 5.3, 5.4, 5.5, and 5.6.

5.3.2 Division Routine

The division routine is depicted in Figure 5.7.

5.3.3 Motion and Direction Change Routines

The motion and direction change routines are illustrated in Figures 5.8, 5.9, and 5.10.

5.3.4 Collision Resolution Routine

The collision resolution routine is illustrated in Figure 5.11.

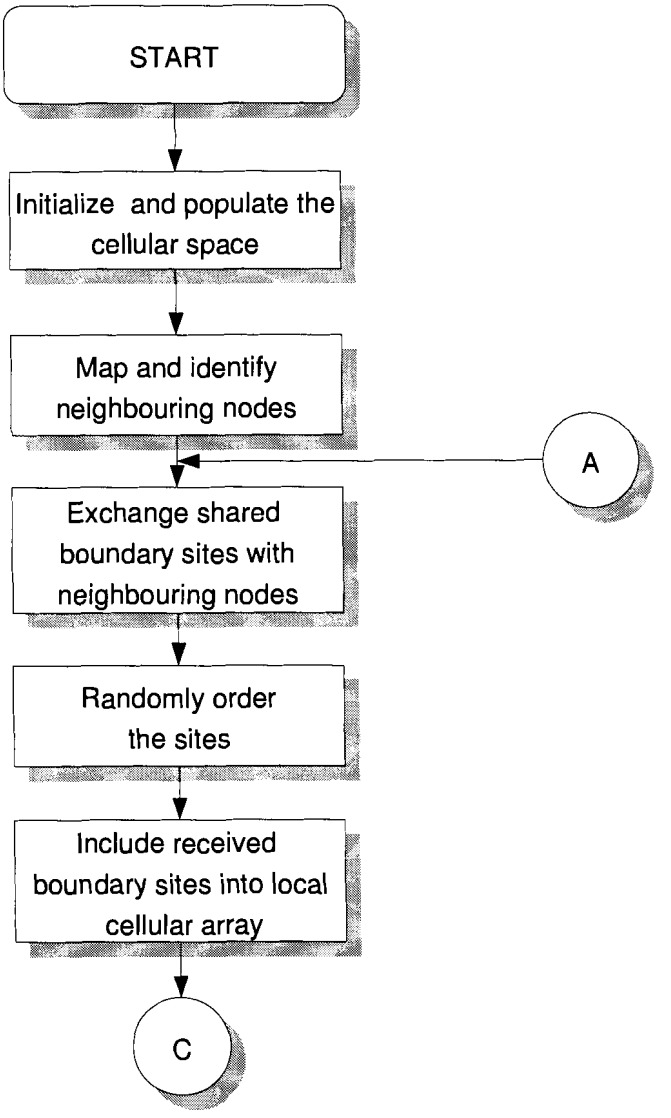


Figure 5.2: Flowchart illustrating the main module of the parallel algorithm (part 1 of 5).

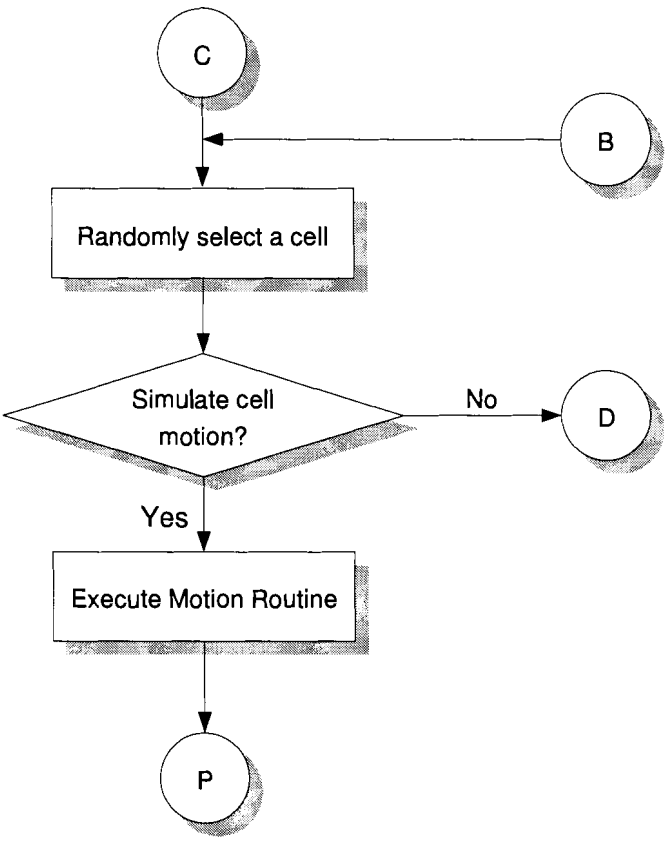


Figure 5.3: Flowchart illustrating the main module of the parallel algorithm (part 2 of 5).

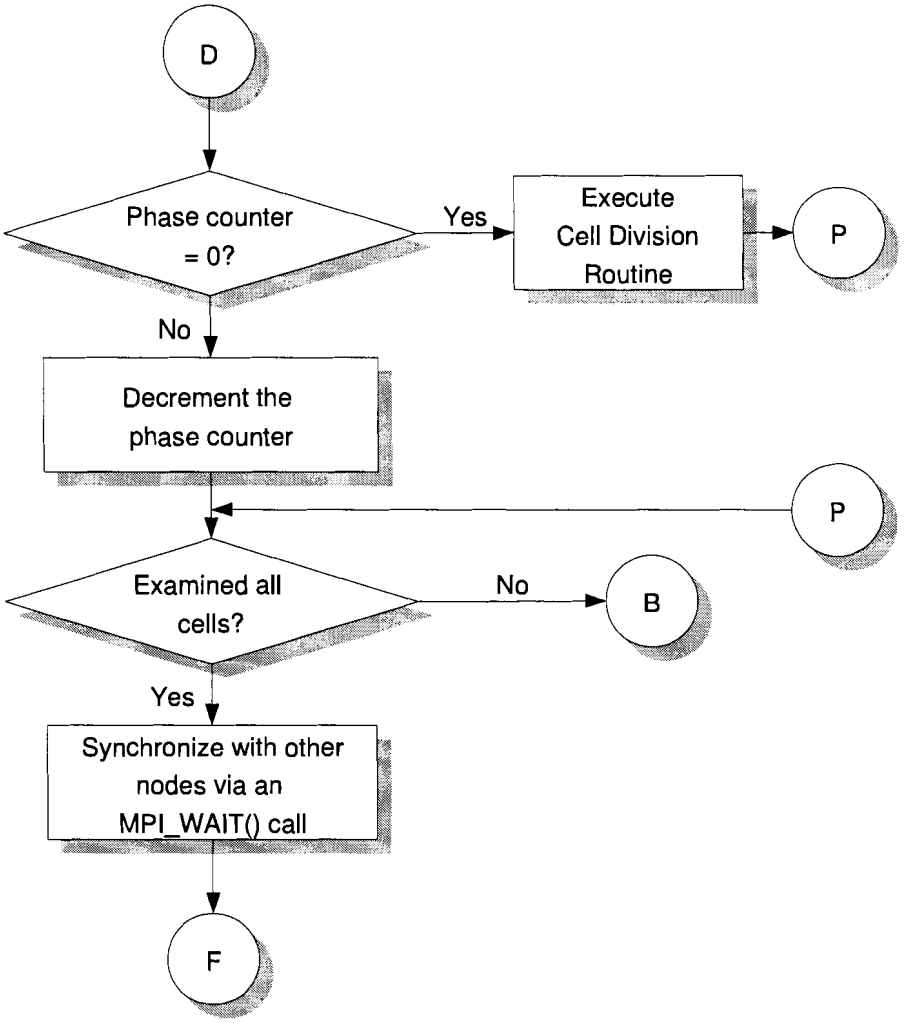


Figure 5.4: Flowchart illustrating the main module of the parallel algorithm (part 3 of 5).

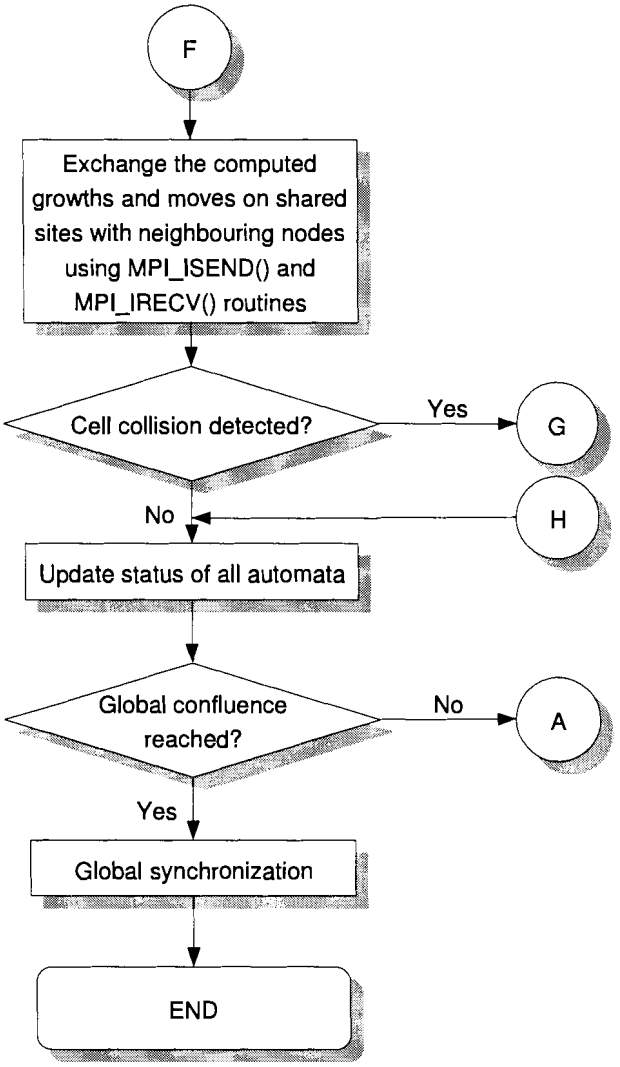


Figure 5.5: Flowchart illustrating the main module of the parallel algorithm (part 4 of 5).

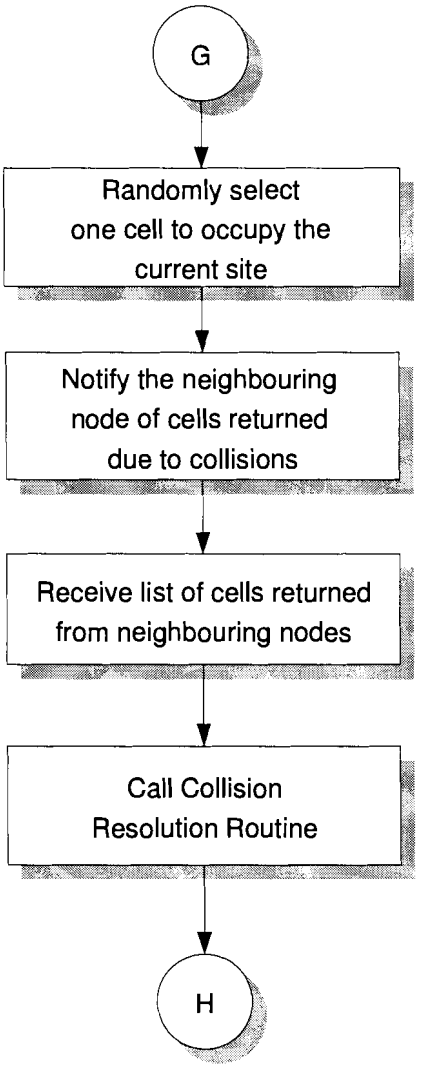


Figure 5.6: Flowchart illustrating the main module of the parallel algorithm (part 5 of 5).

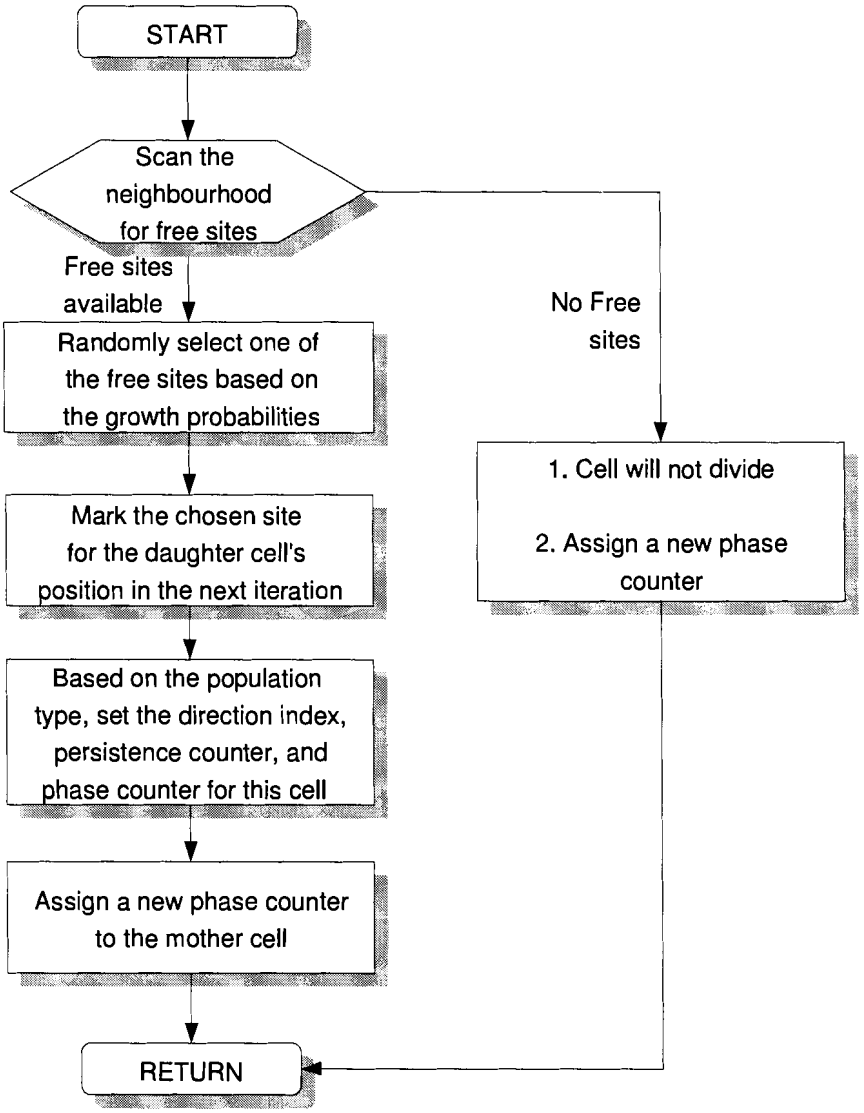


Figure 5.7: Flowchart illustrating the Division Routine of the parallel algorithm.

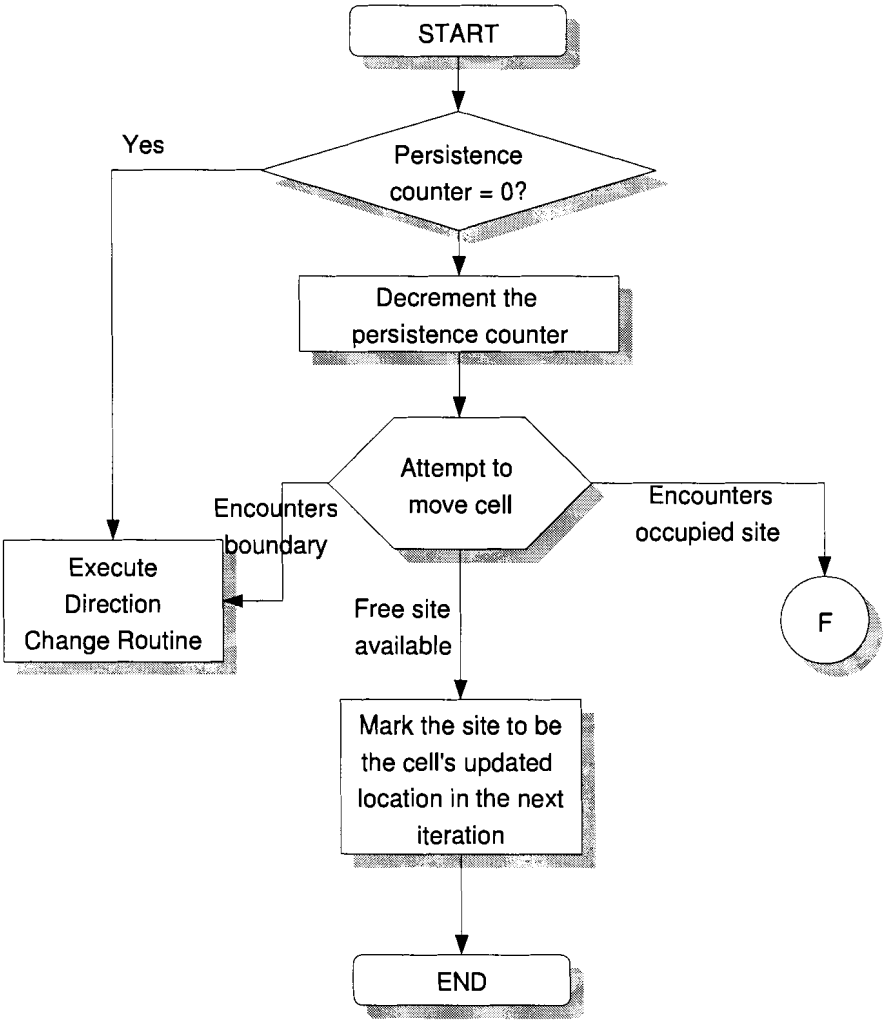


Figure 5.8: Flowchart illustrating the Motion Routine of the parallel algorithm (part 1 of 2).

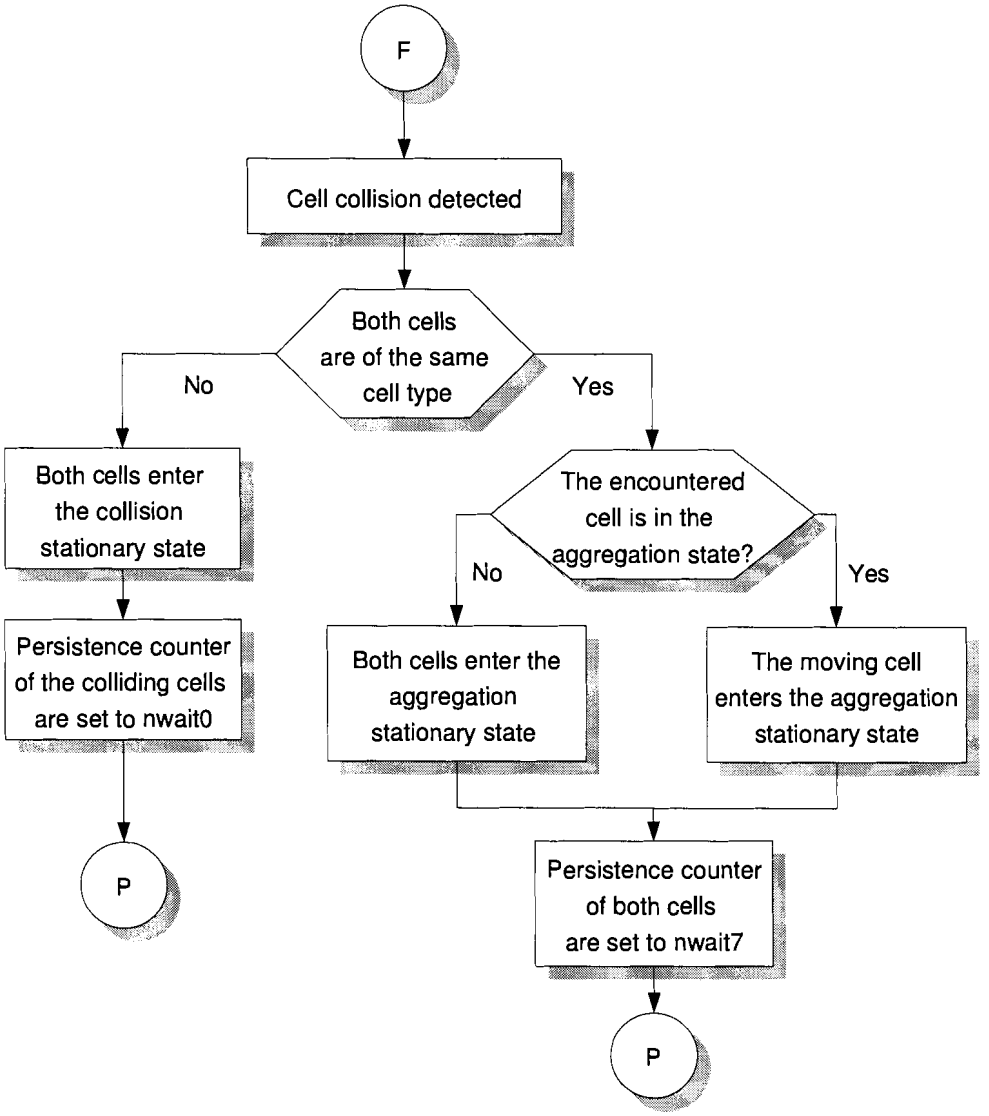


Figure 5.9: Flowchart illustrating the Motion Routine of the parallel algorithm (part 2 of 2).

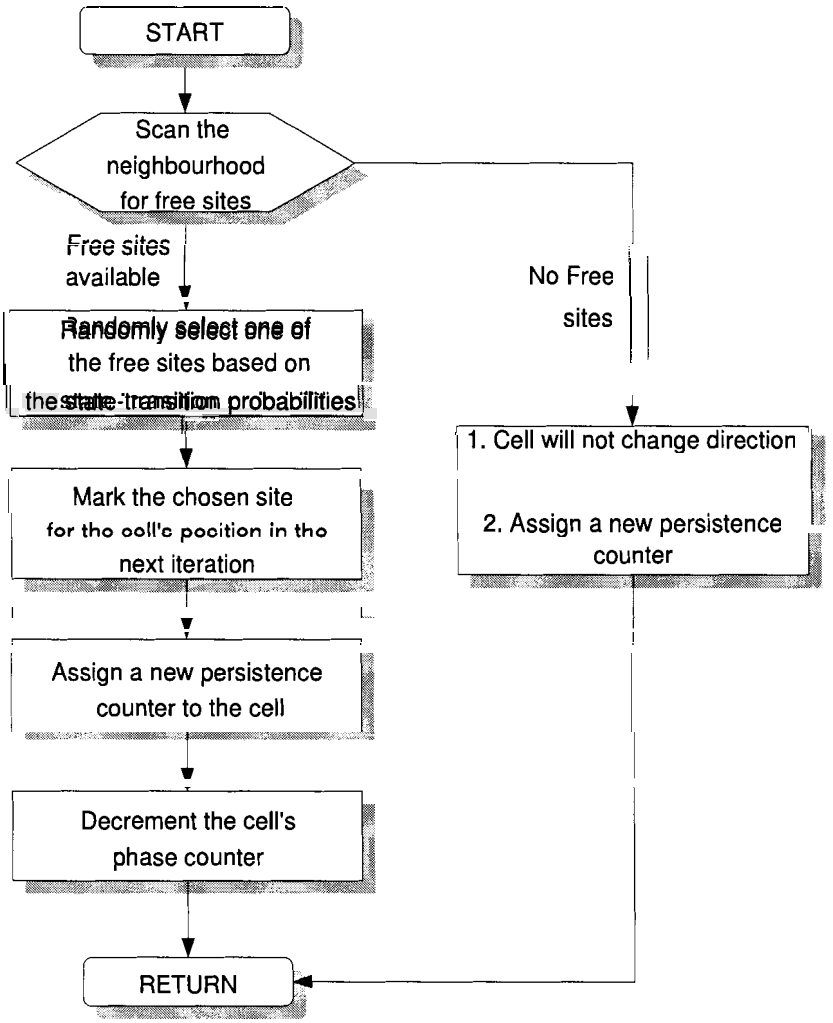


Figure 5.10: Flowchart illustrating the cell Direction Change Routine of the parallel algorithm.

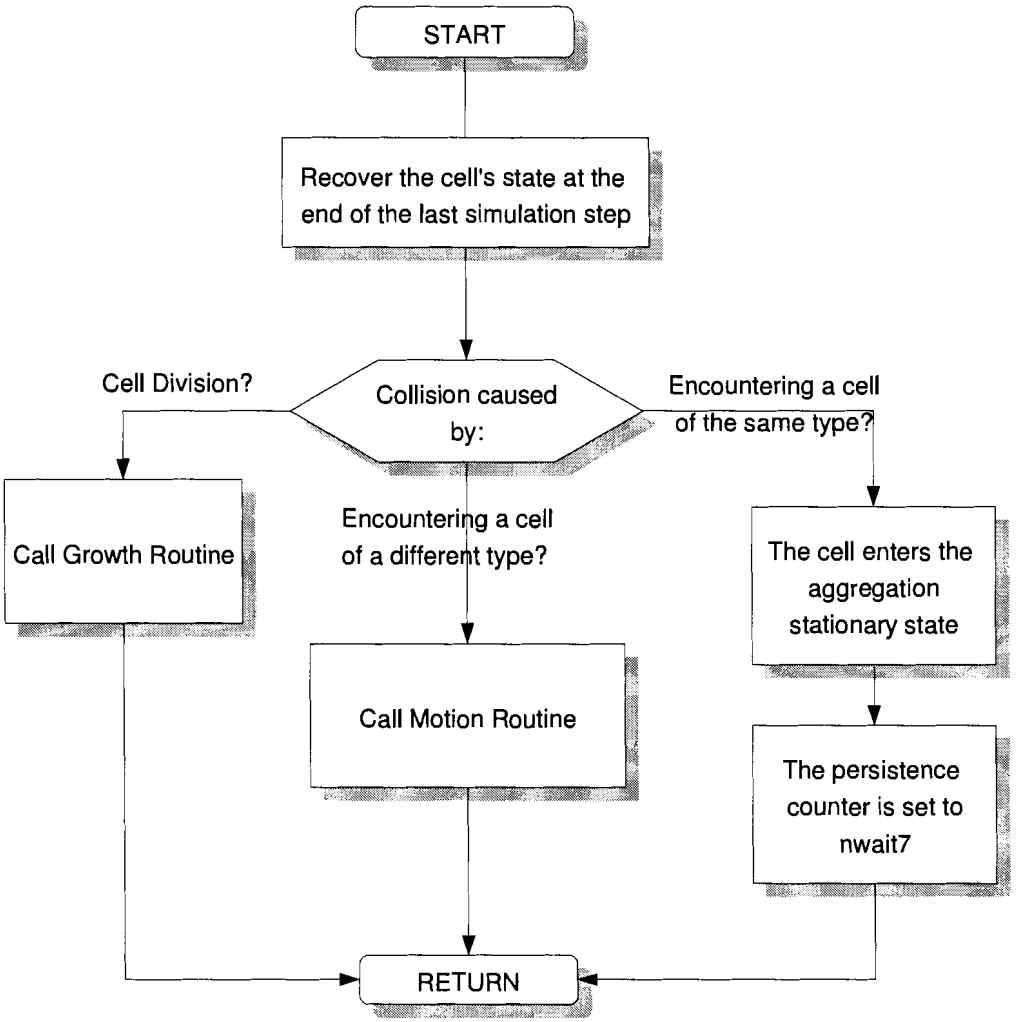


Figure 5.11: Flowchart illustrating the Collision Resolution Routine of the parallel algorithm.

5.4 Load Balancing

Load balancing can be done statically or dynamically. Static load balancing is done at the start of the simulation. Each node is responsible for a fixed part of the problem domain. The advantage of the static method is that the entire overhead of the load-balancing process is incurred at compile time, resulting in higher efficiency. Dynamic load balancing allows for the adaptation to application requirements during run time. However, such strategies incur a run-time overhead due to load information transfer among processors, the decision making process for the selection of processes and processors, and communication delays due to task migration.

We chose static load balancing. The choice is based on the fact that the behaviour of cells is random. The computational load fluctuations within a subdomain tend to average out, thus maintaining a load-balanced computation. The processing of an occupied site is kept local because it depends only on the state of its neighbourhood index. Our parallel algorithm preserves the locality property of the algorithm, which guarantees that all neighbours of a cell in the cellular space are stored either on the same processor, or on a processor that is physically an immediate neighbour to it.

5.5 Parallelization of Random Number Generation

Our choice to use the Lehmer generator for random numbers was based not only on the quality of the generated random numbers but also on its amenability to be parallelized in such a way to ensure the reproducibility of simulation runs and assume an adequate degree of independence of the parallel streams of random numbers.

The generator is defined as:

$$x_{n+1} = ax_n \bmod m, \tag{5.1}$$

where x_n is the nth member of the sequence of random numbers before normalization, m is a large prime number, and a is an integer ranging from 2 and $m-1$. Given Equation (5.1), we can obtain the $(n+k)$th member of the sequence in terms of the nth:

$$x_{n+k} = Ax_n \bmod m, \tag{5.2}$$

where

$$A = a^k.$$

Assume that we have access to a parallel machine consisting of P processors connected by some communication network. The idea is then to have each processor compute random numbers using Equation (5.2) with $k = P$. Since the value of P is fixed for a given run and easily obtainable, the value of A can be computed once and stored. We give the processors a staggered start to prevent their respective sequences from overlapping. Let y_1 denote the seed of the sequential algorithm, chosen as described before. Sequentially and before the normalization step, the following sequence is obtained:

$$\begin{aligned} y_1 &= y_1 \\ y_2 &= ay_1 \bmod m \\ y_3 &= ay_2 \bmod m \\ &\vdots \\ y_n &= ay_{n-1} \bmod m \end{aligned}$$

We set the seeds in the processors of the parallel machine in the following way, where a subscript denotes the position in the random number sequence and a superscript denotes a processing node,

$$\begin{aligned} x_1^{(1)} &= y_1 \\ x_1^{(2)} &= ay_1 \bmod m = y_2 \\ &\vdots \\ x_1^{(P)} &= y_P. \end{aligned}$$

The *staggered* start is thus defined. Each processor then uses Equation (5.2) to calculate the next member of its sequence where k is now replaced with P. Consequently, we have

$$\begin{aligned} x_2^{(1)} &= y_{1+P} \\ x_2^{(2)} &= y_{2+P} \\ &\vdots \\ x_2^{(P)} &= y_{P+P} = y_{2P}, \end{aligned}$$

and

$$
\begin{aligned}
x_3^{(1)} &= y_{1+2P} \\
x_3^{(2)} &= y_{2+2P} \\
&\vdots \\
x_3^{(P)} &= y_{P+2P} = y_{3P},
\end{aligned}
$$

and so on [17]. We illustrate the above method in Figure 5.12, for $P = 4$ processors. We observe from the figure how the start of the processors is staggered and how the nodes *jump* ahead one another in computing the next random number. Hence the name *leaping* method is used. A nearly exact correspondence is seen between the single sequence obtained on a serial machine and the interleaved one on P processors.

The new multiplier is chosen so as to produce large leaps or hops of P through the sequence of random numbers. Computing the multiplier efficiently is made easier thanks to the associativity of the modulo operation with respect to multiplication. The seeds of the P processors can be rewritten as

$$
\begin{aligned}
x_1^{(1)} &= y_1 \\
x_1^{(2)} &= y_2 = ay_1 \bmod m \\
x_1^{(3)} &= y_3 = a^2 y_1 \bmod m = (a^2 \bmod m) y_1 \bmod m \\
&\vdots \\
x_1^{(i)} &= y_i = a^{i-1} y_1 \bmod m = (a^{i-1} \bmod m) y_1 \bmod m \\
&\vdots \\
x_1^{(P)} &= y_P = a^{P-1} y_1 \bmod m = (a^{P-1} \bmod m) y_1 \bmod m
\end{aligned}
$$

Knowing the initial seed of the sequential run, y_1, as well as the values of a, m, and i for $1 \leq i \leq P-1$ allows us to directly compute the above seeds for the parallel run. The latter values of i correspond to processor identification numbers and are easily obtainable via local library functions. Next, each processor i, $1 \leq i \leq P$, computes its own sequence of j random numbers using the following formulae which take advantage of Equation (5.2), with $k = P$,

$$
\begin{aligned}
x_2^{(i)} &= y_{[i+P]} = Ay_i \bmod m = a^P y_i \bmod m = \\
&\quad (a^P \bmod m) y_i \bmod m = (a^P \bmod m) x_1^{(i)} \bmod m,
\end{aligned}
$$

$$
x_3^{(i)} = y_{[i+2P]} = y_{[(i+P)+P]} = a^P y_{[i+P]} \bmod m =
$$

$$(a^P \bmod m)y_{[i+P]} \bmod m = (a^P \bmod m)x_2^{(i)} \bmod m,$$

$$\vdots$$

$$x_j^{(i)} = y_{[i+(j-1)P]} = y_{[(i+(j-2)P)+P]} = a^P y_{[i+(j-2)P]} \bmod m =$$
$$(a^P \bmod m)y_{[i+(j-2)P]} \bmod m = (a^P \bmod m)x_{(j-1)}^{(i)} \bmod m.$$

Thus, we adopt a single generator, a copy of which resides in the local memory associated to each processing node. The parallel generation of random numbers is interleaved in such a way that is reproducible and to avoid memory conflicts. It is based on the number of available processors. Choosing a single generator gives us also the flexibility to select its parameters and to use knowledge gained thus far in determining and understanding its statistical properties.

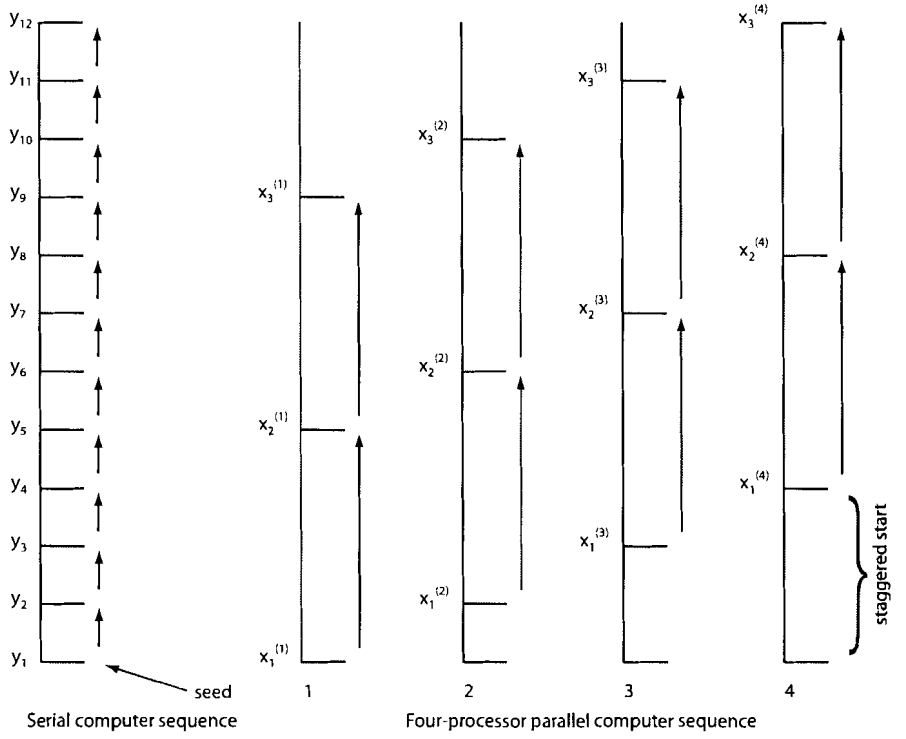


Figure 5.12: A comparison of sequential and parallel random number generation using the leaping method for $P = 4$ processors.

5.6 The Beowulf Cluster

The computing platform used to implement the model, sequentially and in parallel, is the Beowulf cluster, *Nebula*. It is located in the InfoNet Media Centre at Simon Fraser University Surrey. The cluster consists of 128 interconnected nodes using Gigabit Ethernet. Each node uses an Intel P4 3.0-GHz processor with 1 GB of RAM. These nodes run the Gentoo Linux operating system with a GCC compiler version 3.4.4 and a LAM-MPI version 7.1.1.

5.7 Message Passing Interface

The Message Passing Interface (MPI) was selected as the parallel programming specification to implement the parallel algorithm. As an established standard for distributed-memory parallel machines, MPI is a versatile Application Programming Interface (API) that allows for efficient message passing between interconnected nodes [63]. It is based on the Single Program Multiple Data (SPMD) paradigm, where a single program is employed to process multiple pieces of data. MPI provides facilities for interprocessor communication, synchronization, virtual topologies, global reduction operations, timing, and many others [51].

5.8 Chapter Summary

In this chapter, we discussed the steps required for the parallelization of the three-dimensional model on a cluster using the message-passing programming model via MPI. We also presented a sample flowchart for a node program and elaborated on the parallelization of the used random number generator. Lastly, we introduced the computing platform and briefly reviewed the MPI programming specification.

Chapter 6

Simulation Results

6.1 Introduction

Various parameters in the model permit a detailed study of the population dynamics of migrating and proliferating mammalian cells of different types. The flexibility of the model allows us to explore the influence of several system parameters and different cell population characteristics on the tissue growth rate and other aspects of cell behaviour such as the average cell speed and average collisions. The primary parameters of our simulations are the following:

1. Initial volume coverage.
2. Cell motility.
3. Seeding Distribution.
4. The number of cell populations and their population characteristics.

In this thesis, we report results obtained in the case of two cell populations only. We next present the simulation parameters used in our experiments.

6.2 Simulation Parameters

- **Size of the Cellular Array:** Both the serial and parallel simulations were carried out on three-dimensional computational grids containing N^3 sites, where N was varied

from 150 to 330. This value is dictated by the memory capacity of a node. The maximum value of N that we can use also depends on the number of processors. Each occupied computational site is assumed to represent a living mammalian cell of size $10\mu m$ in each dimension.

- **Number of Processors:** Due to limitations of the memory capcity of a single node, we restrict the number of processors P to vary from 2 to 25. Using larger values of P allows us to simulate tissue growth on larger cellular array sizes. However, this makes the comparison of performance results between the sequential and parallel algorithms unfeasible.

- **Confluence:** Confluence is attained at a value of 1.0 indicating 100% volume coverage.

- **Seeding Topologies and Initial Volume Coverage:** Two seeding topologies are examined in the simulations, namely, a uniform topology and a "wound" seeding topology. In the former, the initial volume coverage is varied from 0.1% to 10%. In the latter, all computational sites outside of the denuded wound area are seeded with cells.

- **Seeding Distributions:** The seeding distributions examined include a mixed distribution and a segmented distribution. In the former, the different cell types are uniformly distributed in the cellular space. In the latter, each cell type is seeded in a separate area of the cellular space.

- **Diameter and Height:** These only apply to the "wound" seeding topology. The diameter and height of the cylindrical wound take values from 1 to N.

- **Cell Division Time Distribution:** Two cell types are simulated and each type has a different cell division time distribution. Each division time distribution is associated with a given cell population as follows:

  - **For Cell Population 1** - 64% of the living cells have division times between 12 and 18 hours, 32% of the living cells have division times between 18 and 24 hours, and 4% of the living cells have division times between 24 and 30 hours.
  - **For Cell Population 2** - 4% of the living cells have division times between 12 and 18 hours, 32% of the living cells have division times between 18 and 24 hours, and 64% of the living cells have division times between 24 and 30 hours.

The cell division time distributions for the two populations of cells are summarized in the Table 6.1.

Table 6.1: Cell Division Time Distributions for the Two Cell Populations.

	Cell Populations	
Division Times (hrs)	Cell Population 1	Cell Population 2
$[12, 18)$	64%	4%
$[18, 24)$	32%	32%
$[24, 30)$	4%	64%

- **Time Step:** The value for each time step is varied based on the speed of cell locomotion.
- **Average Waiting Times:** These define the average number of time steps that a migrating cell can spend in each of the six directions and the two stationary states.
- **Growth Probabilities:** They indicate the probability that a cell will divide and the daughter cell will occupy the empty computational site if it is the only free site in the neighbourhood.
- **State-Transition Probabilities:** They denote the likelihood of a cell in a directional state j will enter state k (k, j = 0, 1, 2, ... 7). When a cell's persistence counter reaches zero, the state-transition probabilities are used to determine the next direction of motion.
- **Cell Heterogeneity:** We define the heterogeneity measure, H, as the ratio of the initially seeded number of cells from population 1 to that from population 2. The ratio H is given by:

$$H = \frac{\text{initial number of cells from population 1}}{\text{initial number of cells from population 2}}.$$

That is, when $H = 9$, there are 9 cells from population 1 for every cell from population 2 in the cell seeding density. In our simulations, cells of population 1 are the faster moving cells, while the cells of population 2 are the slower moving cells.

6.2.1 Cell Population Dynamics

Starting with a total number of seed cells equal to N_0, the cellular automata rules described in Chapter 3 transform the cellular array to simulate the dynamic process of tissue growth. At some time t after the start of the simulation, $N_c(t)$ sites of the cellular automaton are occupied by cells. A measure to indicate the volume coverage at time t is the cell volume fraction $k(t)$ defined below:

$$k(t) = \begin{cases} \dfrac{N_c(t)}{N_t} & = \dfrac{\sum_{i=1}^{n} N_{c_i}(t)}{N_t}, \text{ for a uniform seeding topology} \\ \dfrac{N_c(t) - N_0}{N_t - N_0} & = \dfrac{\sum_{i=1}^{n} [N_{c_i}(t) - N_{c_i}(0)]}{N_t - \sum_{i=1}^{n} N_{c_i}(0)}, \text{ for a wound seeding topology} \end{cases}$$

where N_t is the size of the cellular space ($= N_x \times N_y \times N_z$), $N_{c_i}(t)$ is the number of occupied computational sites by cell type i at time t, $N_{c_i}(0)$ is the number of seed cells of type i, and n is the number of cell types ($n \geq 1$). For the uniform seeding, the cell volume fraction inidcates the fraction of cells occuping the cellular space and in the wound seeding, it indicates the fraction of cells occupying the wound area.

The overall tissue growth rate is the increase in volume coverage with respect to time. To compute this measure, we use the following two formulae:

$$\frac{dk(t)}{dt} = \begin{cases} \dfrac{\sum_{i=1}^{n} [N_{c_i}(t) - N_{c_i}(t - \Delta t)]}{\Delta t \times N_t}, \\ \dfrac{\sum_{i=1}^{n} [N_{c_i}(t) - N_{c_i}(t - \Delta t)]}{\Delta t \times (N_t - N_0)} & = \dfrac{\sum_{i=1}^{n} [N_{c_i}(t) - N_{c_i}(t - \Delta t)]}{\Delta t \times (N_t - \sum_{i=1}^{n} N_{c_i}(0))}, \end{cases}$$

for a uniform topology and a wound seeding topology, respectively. Here, $k(t)$ is the cell volume fraction at time t as given above and Δt is the time step in days. In the uniform seeding topology, the tissue growth rate applies to the entire cellular space; while in the wound seeding topology, it appiles to the wound area only.

The simulation continues until all sites are occupied by cells, that is until $k(t)$ equals one. The movement of cells will slow down due to breaks in the persistent random walks, cell collisions, and cell aggregations. Thus only a fraction of the cells $N_c(t)$ will move in the time interval $[t, t + \Delta t]$ and the effective speed of migration, S_e, in uniform seedings can be calculated as:

$$S_e(t) = \frac{N_m(t)}{N_c(t)} \times S,$$

where $N_m(t)$ refers to the number of cells that were moving in the time interval $[t, t + \Delta t]$ and S is the cell "swimming" speed. For wound seeding runs, we need to count the number of migrating cells, $N_{m,w}(t)$, located inside the cylindrical wound. The effective speed of migration, $S_{e,w}(t)$, is then obtained via:

$$S_{e,w}(t) = \frac{N_{m,w}(t)}{N_{c,w}(t)} \times S,$$

where $N_{c,w}(t)$ is the number of occupied computational sites in the wound area at time t.

Below, we present the simulation results from both the serial and parallel implementations of the model. These results may have significant implications for the design of the laboratory experiments aimed at studying the roles of the previously mentioned parameters on the rates of tissue growth.

6.3 Serial and Parallel Results

The simulations for the sequential and parallel implementations were obtained using a $200 \times 200 \times 200$ cellular array, a 99.99% confluence parameter, and average waiting times of 2 hours for the six directional states and 1 hour for the two stationary states, respectively. In the wound seeding topology, the empty cylinder has a diameter of 100 and a height equal to the height of the cellular array of 200. Unless specified otherwise, the migration speeds of cells in populations 1 and 2 are equal to 10 $\mu m/hr$ and 1 $\mu m/hr$, respectively. In the parallel simulations, the results were obtained using the slab decomposition technique for both the uniform and wound seeding topologies. Eight processors of the cluster were utilized in this case. This means that each node processed a local cellular array of size $200 \times 200 \times 25$. All remaining parameters were kept the same as for the serial simulations. We note that all the parallel results are similar to the serial ones and hence the same discussion applies.

Our objective in this work is not to evaluate the accuracy of the parallel results, but to obtain such results that behave macroscopically the same way as the sequential ones. As the parallel results will show shortly, we believe that we have achieved this goal rather consistently.

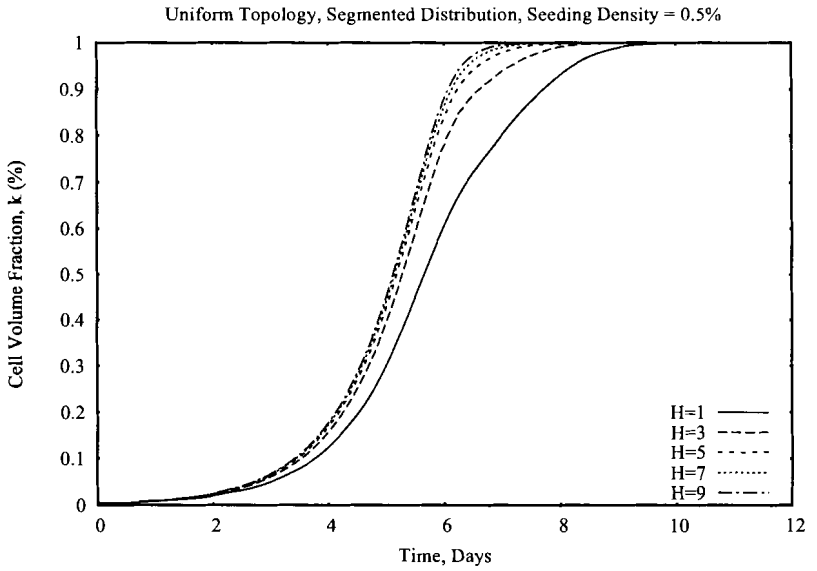
6.3.1 Uniform Seeding Topology

Segmented Distribution

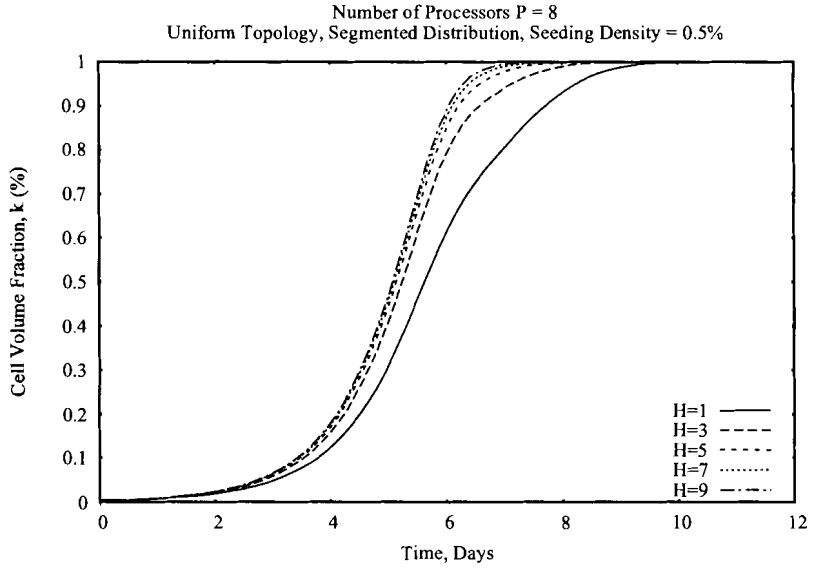
Effects of varying the ratio H

Figure 6.1 shows the cell volume fraction versus time for different values of the ratio H. This ratio indicates a measure of heterogeneity in the initial cell distribution. The volume coverage increases with time until it reaches confluence. In these simulation runs, H is equal to 1, 3, 5, 7 and 9 while the total initial seeding density is maintained at 0.5%. For instance, with $H = 9$ the faster moving population of cells initially are seeded in 90% of the cellular space while the slower moving population are seeded in the remaining 10%. We observe that as H increases, the time taken to reach confluence decreases. This is because for larger values of H, the population of faster moving cells dominates the proliferation. This is clearly depicted in Figure 6.2 and Figure 6.3 where a higher tissue growth rate is observed as H is increased. Faster moving cells spread out in the cellular space preventing the formation of cell colonies; thus allowing for confluence to be reached sooner.

Figure 6.4 and Figure 6.5 show the average number of collisions per hour versus time for cell population 1 and cell population 2 for various values of H, respectively. For cell population 1, as H is increased the average number of cell collisions increases and reaches a higher peak value due to an increase in the number of fast cells in its segment of the cellular space, which leads to more cell-to-cell interactions. Because cells in population 2 have a lower speed of 1 $\mu m/hr$, they tend to form more localized clusters. When combined with the fact that an increase in the ratio H also reduces the size of the segment for this population (and the number of slow cells), this results in fewer and fewer collisions as H is increased. The latter results are depicted in Figure 6.5.

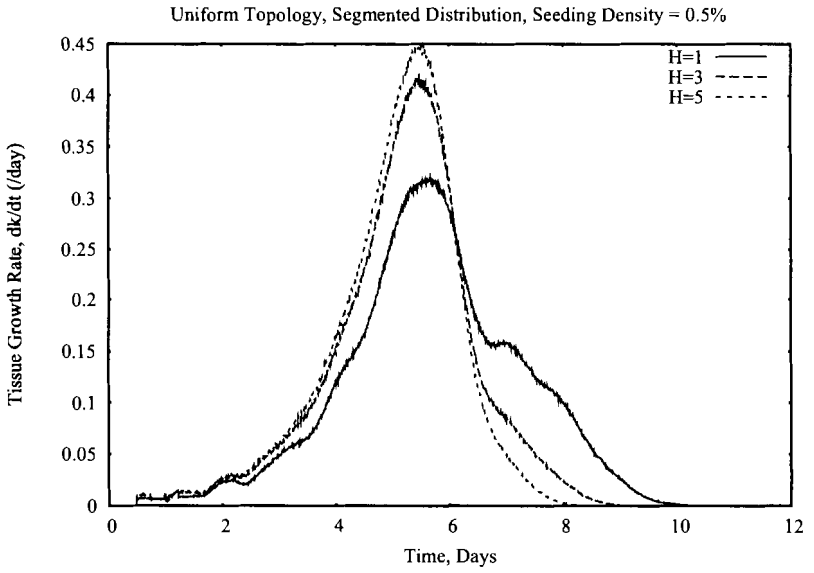


(a) Sequential Results

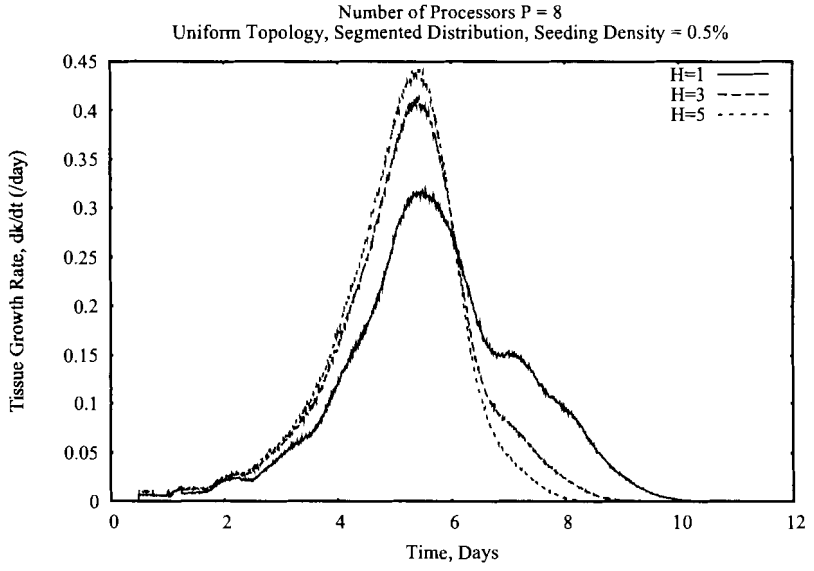


(b) Parallel Results

Figure 6.1: The effects of varying the ratio H on the cell volume fraction.

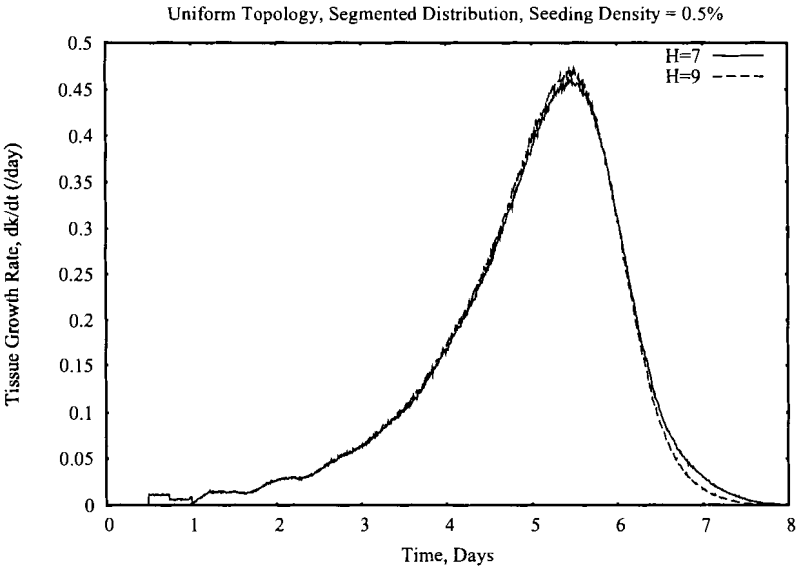


(a) Sequential Results

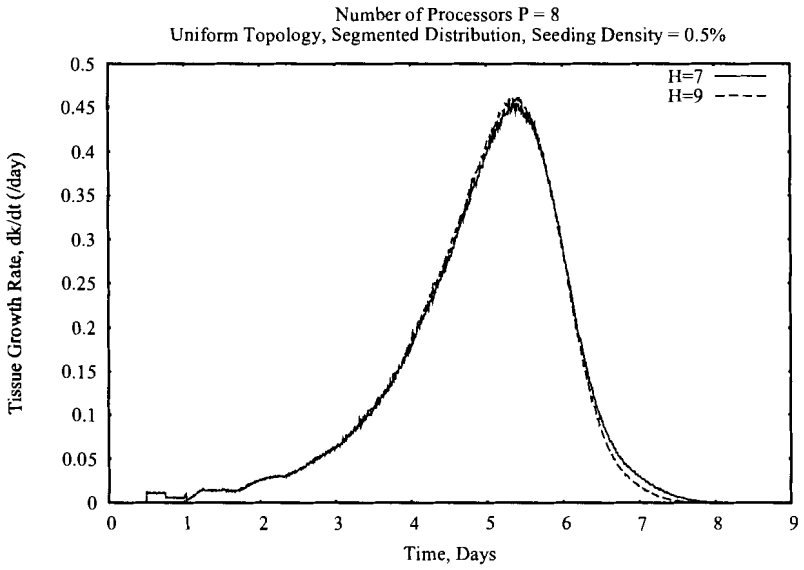


(b) Parallel Results

Figure 6.2: The overall tissue growth rate as the ratio H is varied from 1–5.

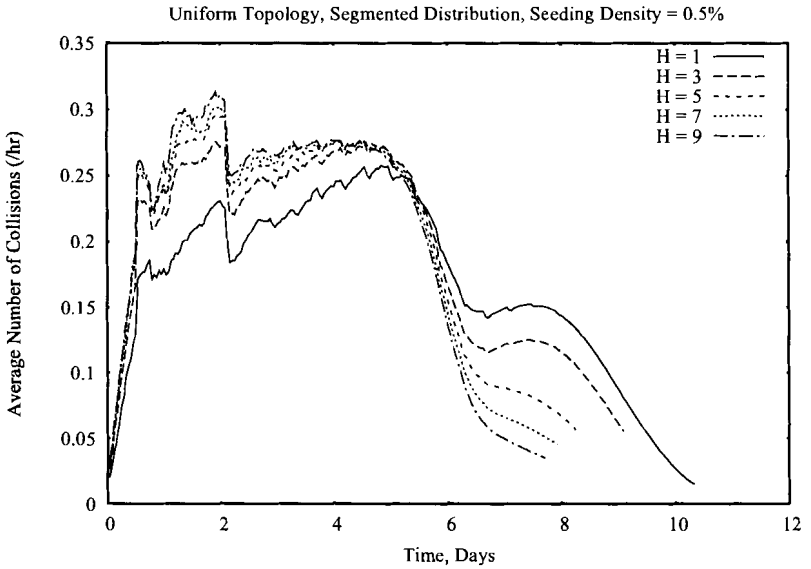


(a) Sequential Results

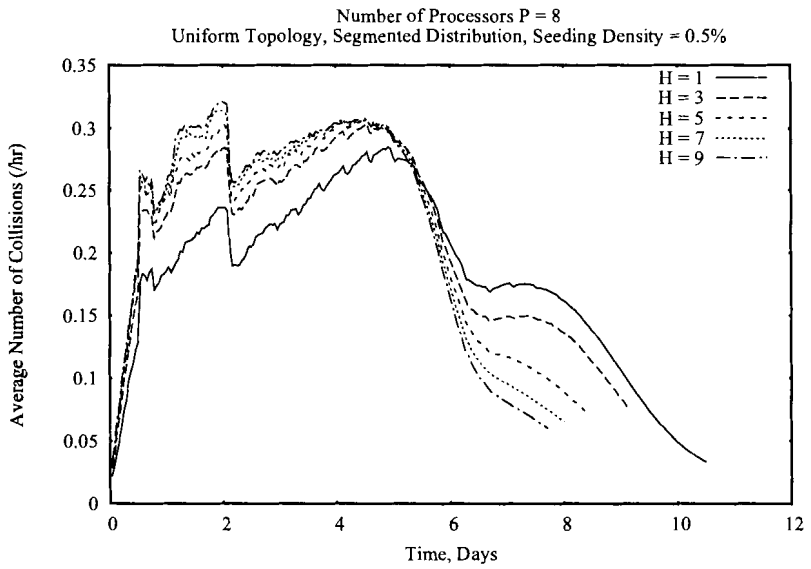


(b) Parallel Results

Figure 6.3: The overall tissue growth rate as the ratio H is varied from 7–9.

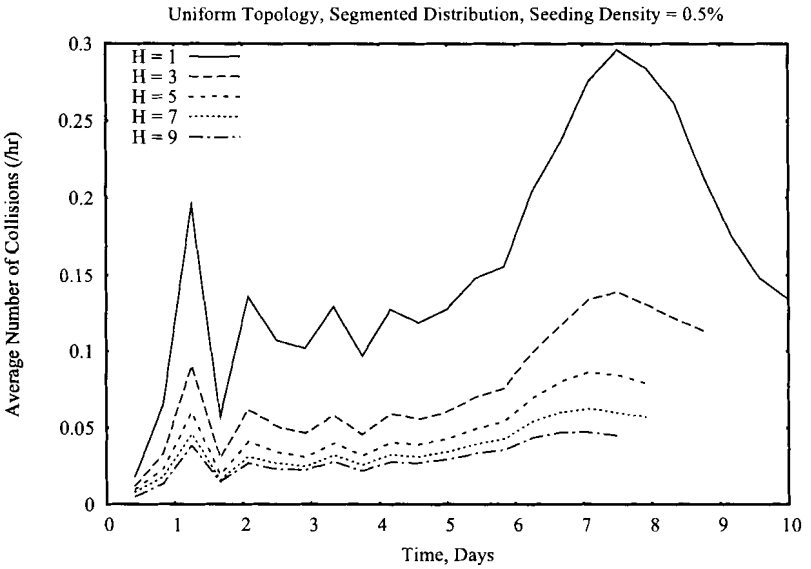


(a) Sequential Results

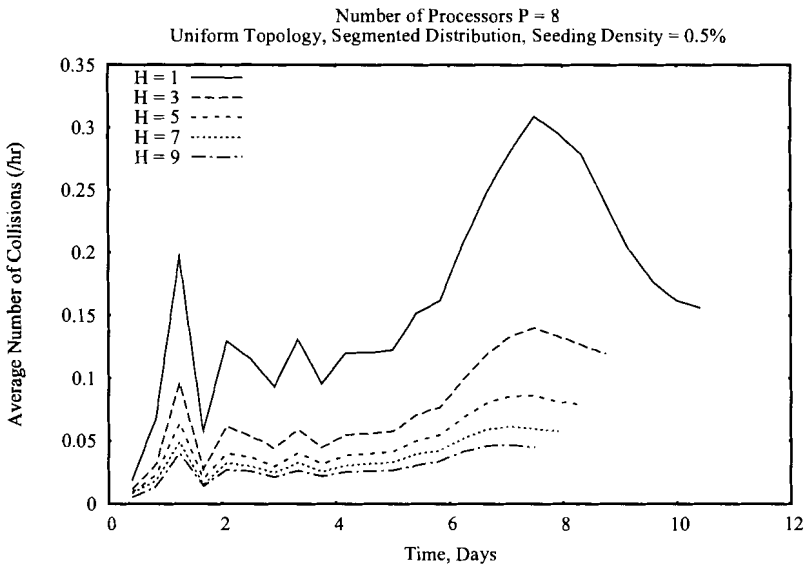


(b) Parallel Results

Figure 6.4: The effects of varying the ratio H on the average number of collisions per hour for cell population 1.



(a) Sequential Results

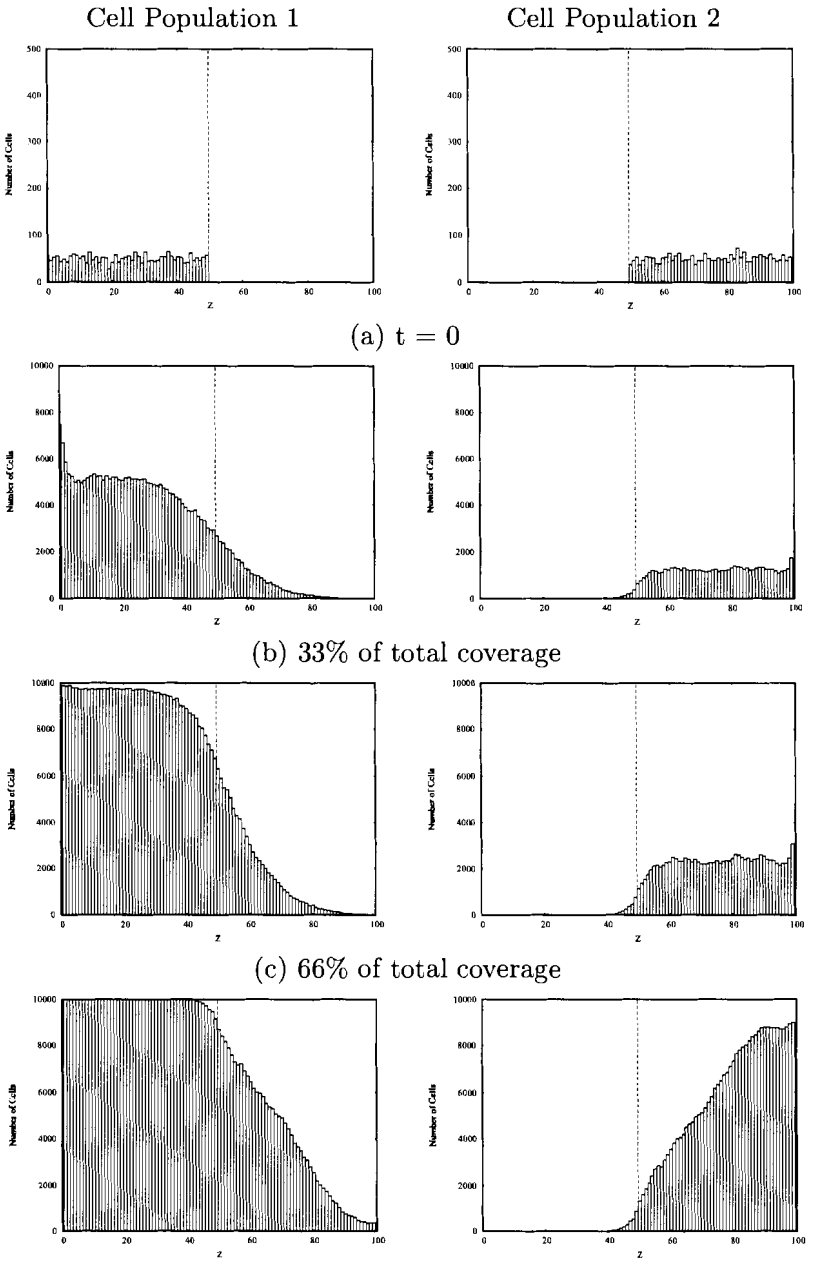


(b) Parallel Results

Figure 6.5: The effects of varying the ratio H on the average number of collisions per hour for cell population 2.

Cell distribution profiles at different time steps

In order to observe the growth of both cell populations in the uniform segmented distribution, we view their profiles on an (x, y) plane at different time steps ranging from 33% of confluence to 99% of confluence as a function of the third dimension (z-axis). A ratio H of 1 and an initial seeding density of 0.5% are utilized as inputs. We observe in Figure 6.6 that initially (at $t = 0$), the two populations are located in their respective areas of the cellular space: the faster moving cells of population 1 in the lower half and the slower moving cells of population 2 in the upper half of the cellular array. As the cells proliferate, both profiles change over time. We observe that the faster cells start diffusing into the area initially occupied by the slower cells whereas the slower cells tend to mostly remain in their area. As confluence is reached, the faster cells completely cover the lower half of the cellular space and also penetrate into the upper half where the slower cells reside. For readability purposes, the presented cell distribution profiles were obtained for a $100 \times 100 \times 100$ cellular array.



(a) t = 0

(b) 33% of total coverage

(c) 66% of total coverage

(d) 99% of total coverage

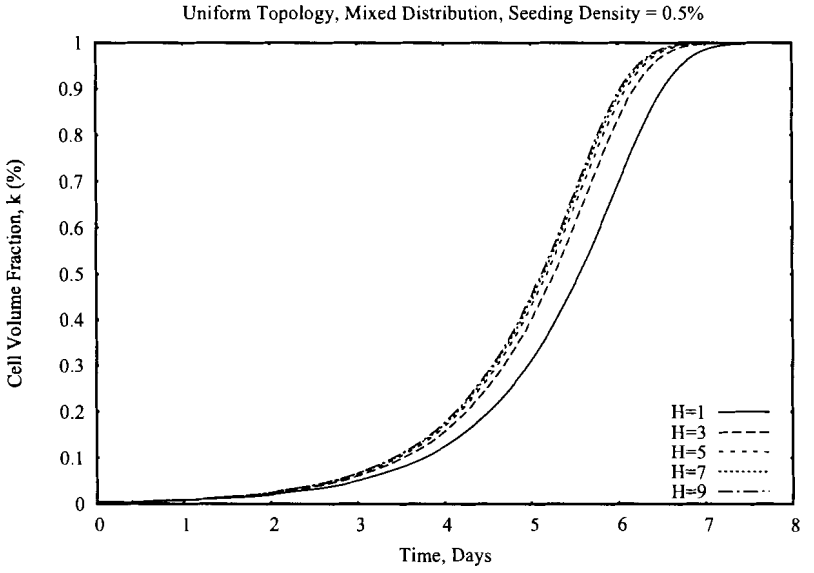
Figure 6.6: Profiles of cell populations 1 and 2 at different confluence values for $H = 1$ and a seeding density of 0.5%. Cells in populations 1 and 2 move at speeds of 10 μm/hr and 1 μm/hr, respectively. A dotted vertical line is included to highlight the diffusion of each cell type from one half of the cellular space to the other.

Mixed Distribution

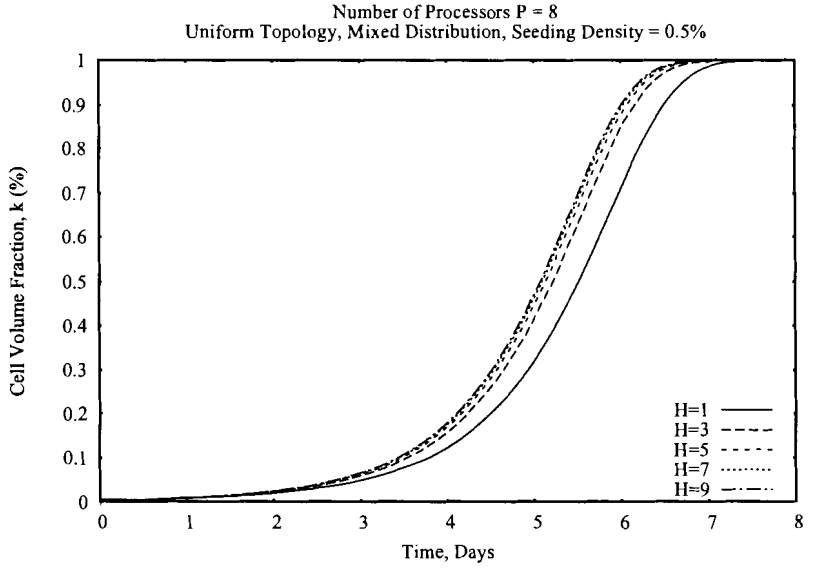
Effects of varying the ratio H

The effects of varying the ratio H on the cell volume fraction and the tissue growth rate in a mixed distribution were also studied and are shown in Figure 6.7 and Figure 6.8, respectively. In these simulations, the total seeding density is 0.5%. We observed similar results to the ones obtained when using a segmented distribution, that is increasing H yields both a decrease in the time to reach complete volume coverage and an increase in the overall tissue growth rate. Faster moving cells spread out in the cellular space preventing the formation of local clusters and hence have faster proliferation as H increases.

Figure 6.9 and Figure 6.10 show the average number of collisions for populations 1 and 2, respectively, for different values of H given a fixed cell seeding density of 0.5%. For cells in population 1, during the first two days and as H is increased, the average number of collisions per hour increases due to an increase in cell-cell interactions between fast and slow moving cells early in the proliferation process. Figure 6.11 and Figure 6.12 clearly illustrate this phenomenon. These two figures depict not only the total average number of collisions but also the average number of collisions between cells of the same type (or homotypic) as well as between cells of different types (or heterotypic) for $H = 1$ and $H = 9$, respectively. After the first two days, this component of the average number of collisions decreases rapidly as H is increased. Given that the average number of collisions between fast cells of the same type increases with time until it reaches a maximum for all values of H, the combined effect of these two observations leads to a reversal of behaviour. That is, the average number of collisions per hour decreases as H is increased after the first two days for cells in population 1. For cells in population 2, the average number of collisions decreases as H is increased due to the decrease in the seeded number of slow cells. This results in a smaller number of cell clusters, which in turn impacts negatively the number of cell-cell interactions between slow cells.

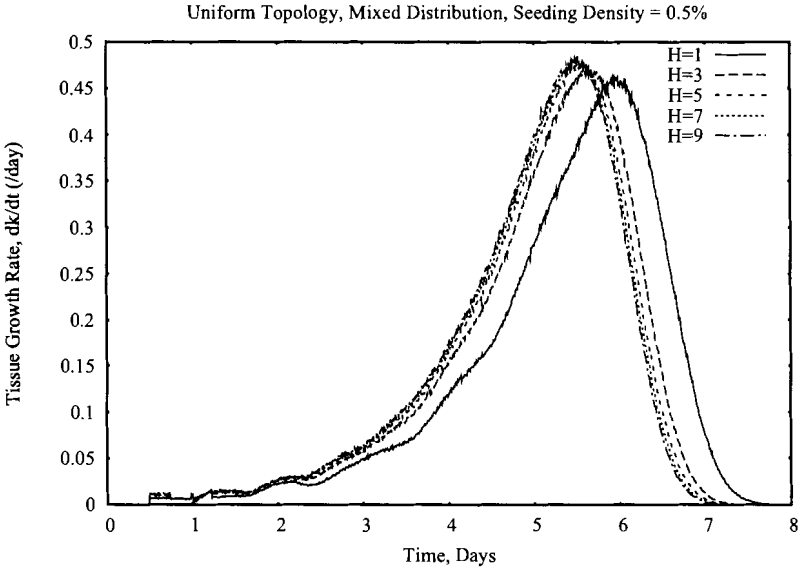


(a) Sequential Results

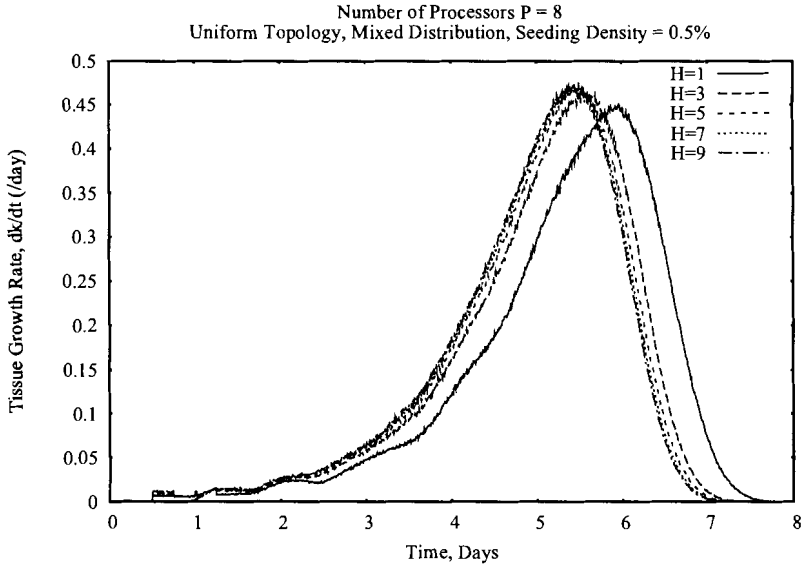


(b) Parallel Results

Figure 6.7: The effects of varying the ratio H on the cell volume fraction.

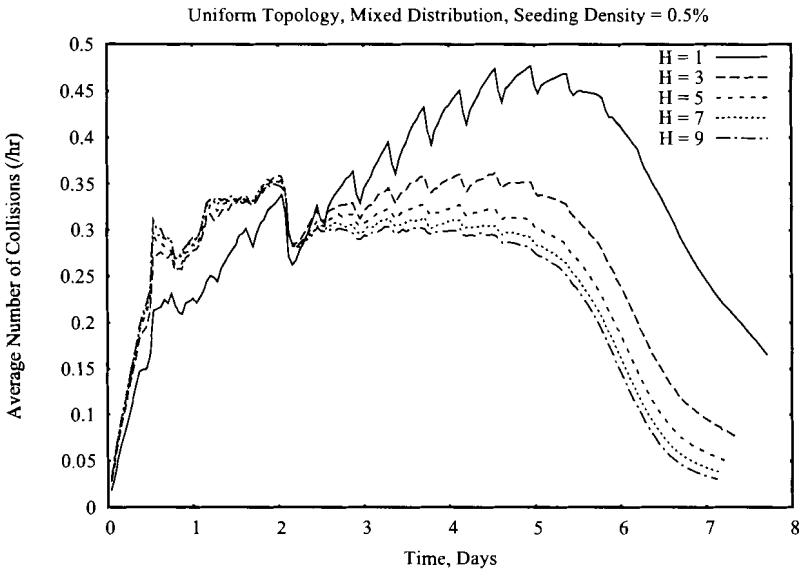


(a) Sequential Results

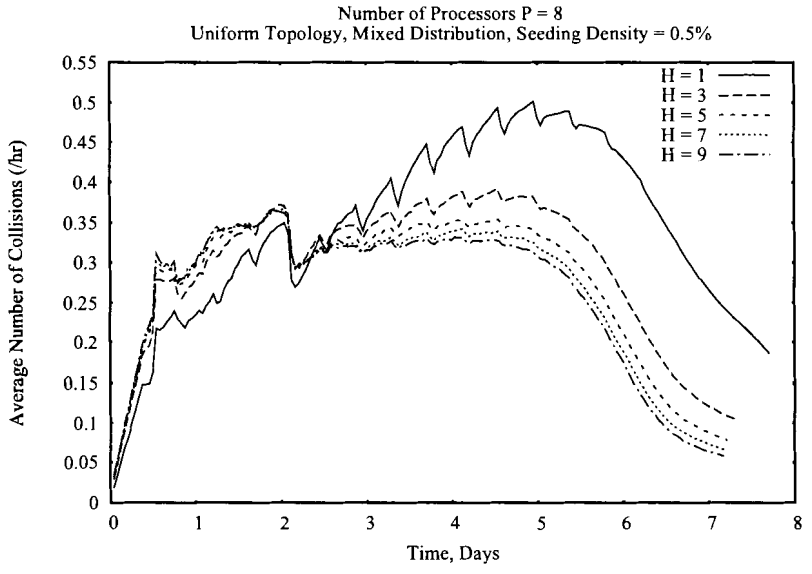


(b) Parallel Results

Figure 6.8: The effects of varying the ratio H on the overall tissue growth rate.

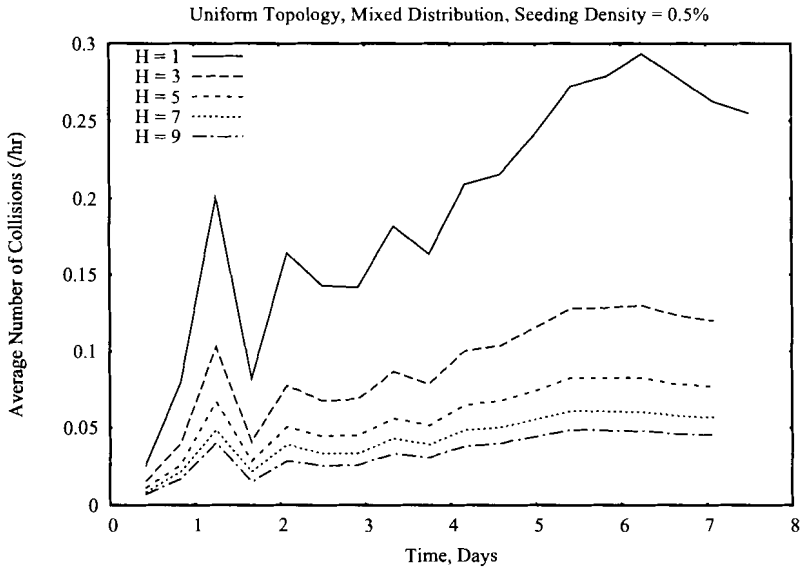


(a) Sequential Results

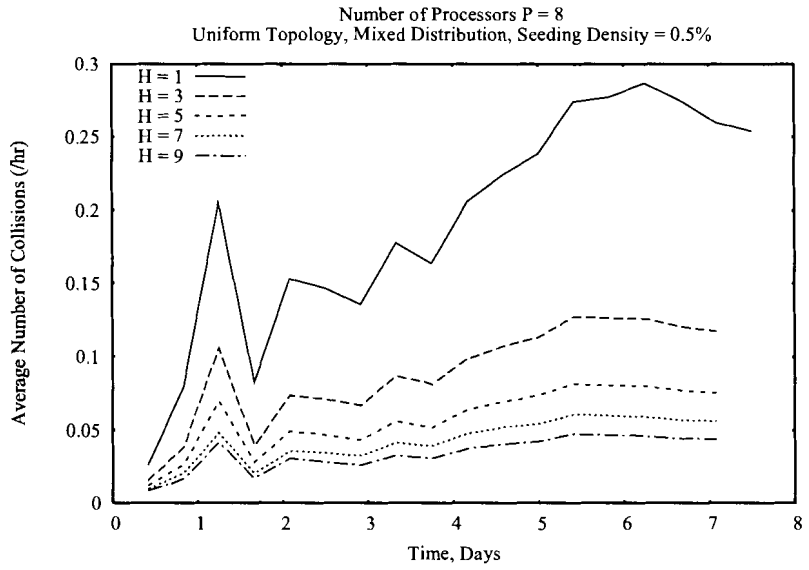


(b) Parallel Results

Figure 6.9: The effects of varying the ratio H on the average number of collisions per hour for cell population 1.

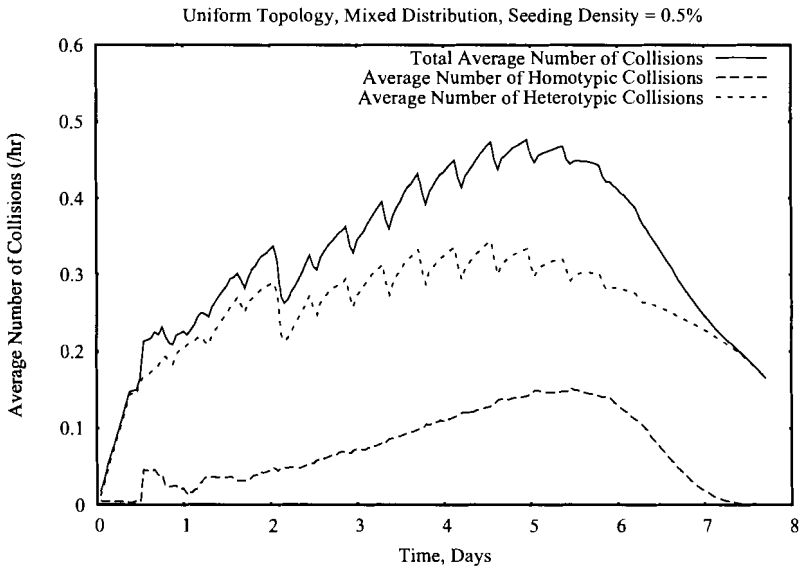


(a) Sequential Results

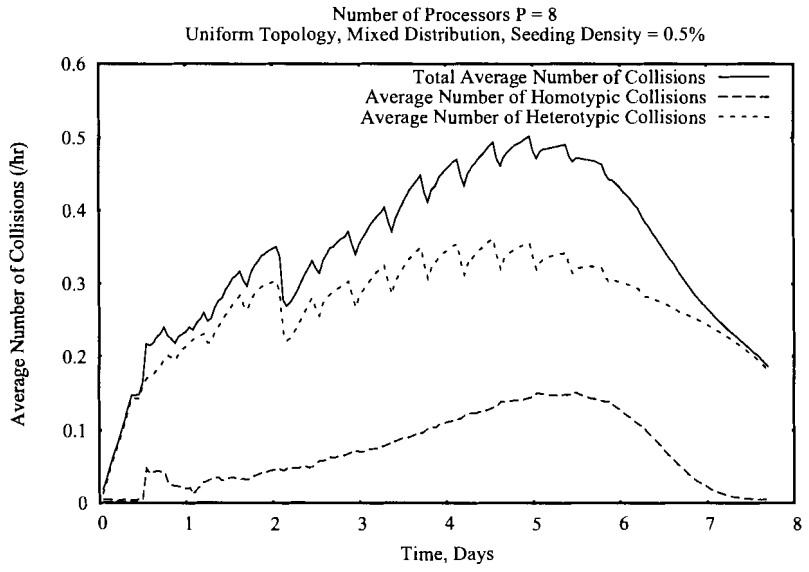


(b) Parallel Results

Figure 6.10: The effects of varying the ratio H on the average number of collisions per hour for cell population 2.

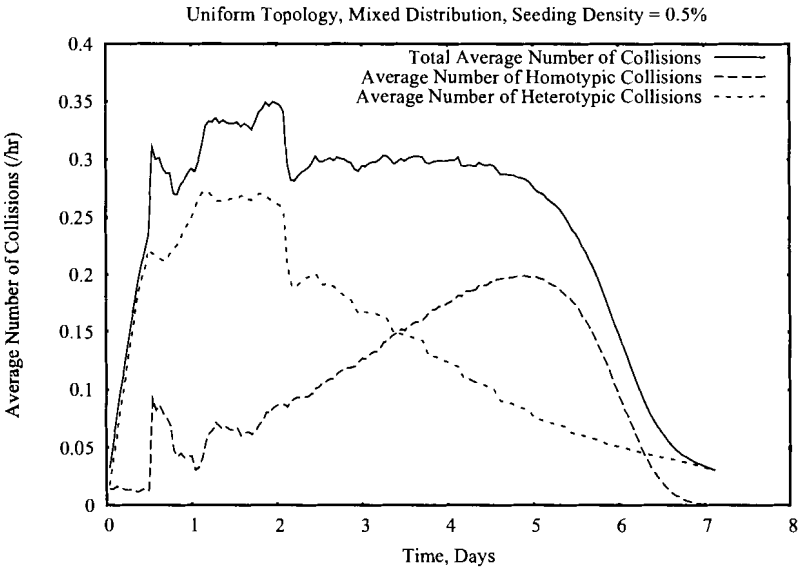


(a) Sequential Results

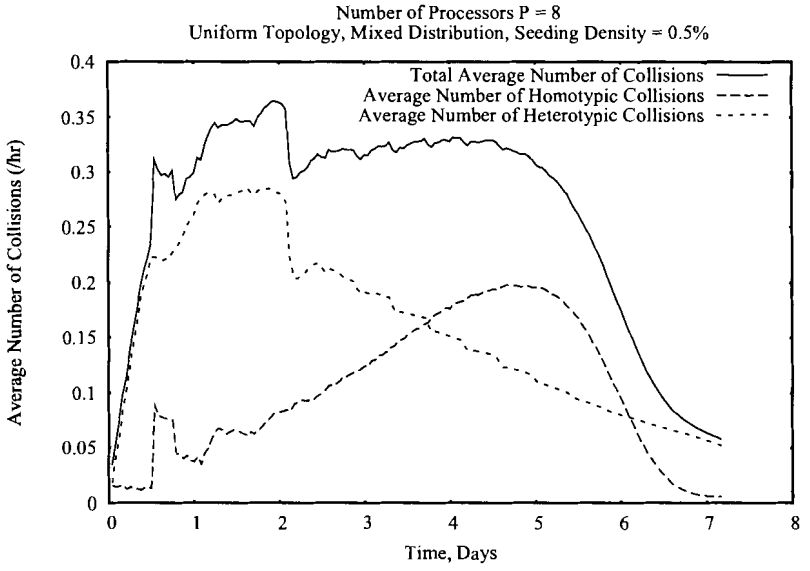


(b) Parallel Results

Figure 6.11: The temporal evolution of the average number of collisions per hour for cell population 1 and its two components for $H = 1$.



(a) Sequential Results

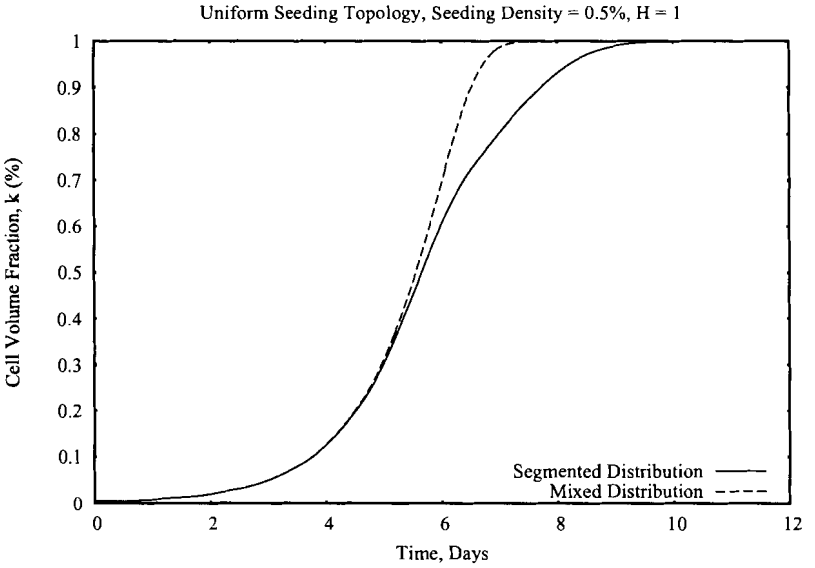


(b) Parallel Results

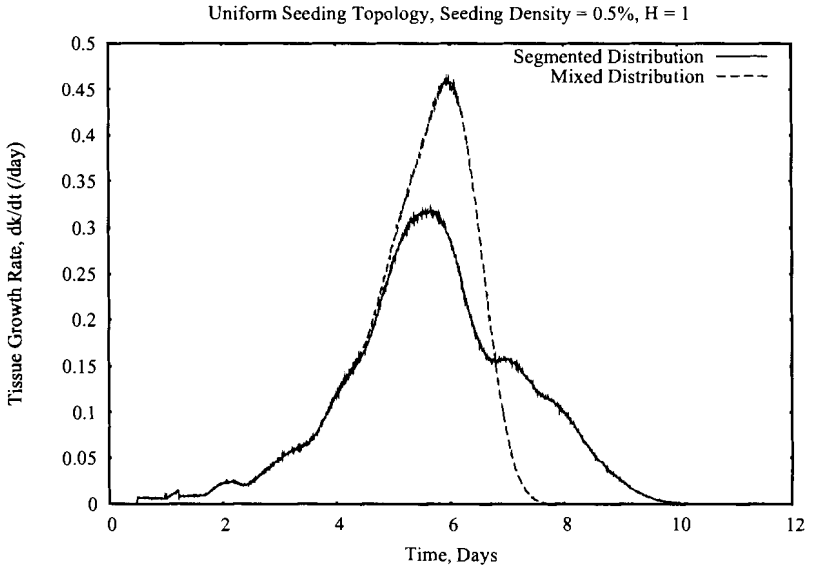
Figure 6.12: The temporal evolution of the average number of collisions per hour for cell population 1 and its two components for $H = 9$.

6.3.2 Comparison of Uniform Cell Seeding Distributions

Figures 6.13 and 6.14 show comparisons between the cell volume fraction and tissue growth rates obtained by using the segmented and mixed uniform seeding distributions. Here, two different values of the ratio H were used, $H = 1$ and $H = 9$, while the total seeding density is 0.5%. In Figure 6.13, we observe that when $H = 1$ the mixed distribution takes less time to reach full volume coverage and yields a higher tissue growth rate. The mixed distribution yields a higher maximum value of the tissue growth rate than the segmented one (0.46 versus 0.32). This may be attributed to the fact that contact inhibition has less of an effect in the mixed distribution where faster cells have more nearby empty spaces to move and divide into which in turn frees up sites for the slower-moving cells as well. Increasing H to 9 shows a stronger positive impact on the time to reach confluence and tissue growth rate in the case of the segmented distribution than the mixed one. The increased number of faster moving cells allows them to disperse in the cellular space and dominate the proliferation, resulting in similar overall tissue growth behavior for both distributions. In this case, the distinction between the two seeding distributions is less apparent.

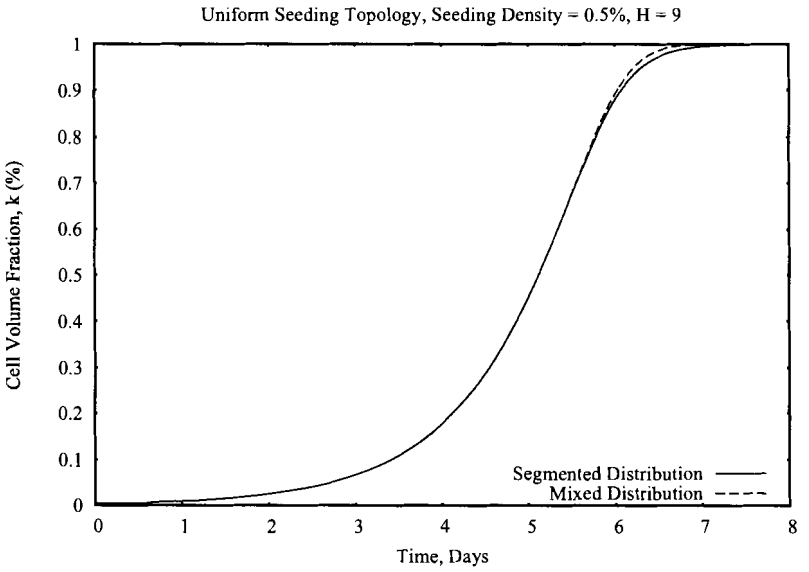


(a)

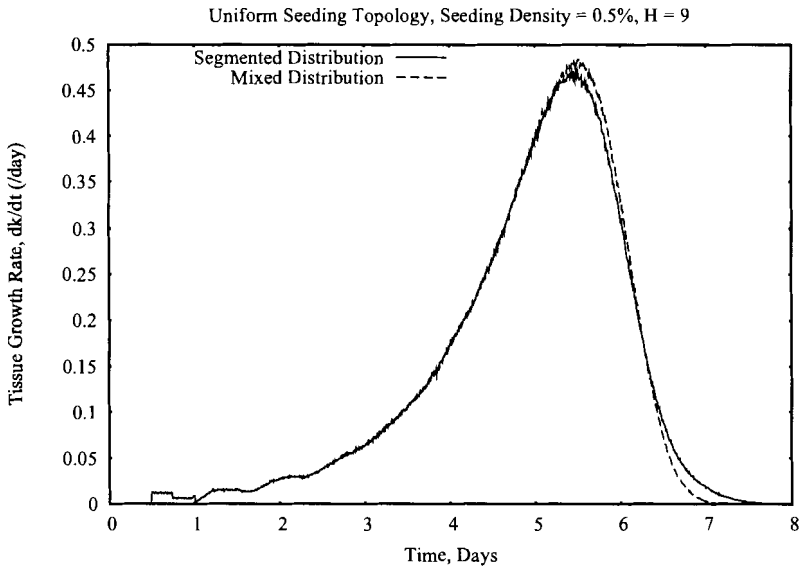


(b)

Figure 6.13: Comparison of (a) the cell volume fraction and (b) the overall tissue growth rate for segmented and mixed seeding distributions. Here, the ratio H is equal to 1.



(a)



(b)

Figure 6.14: Comparison of (a) the cell volume fraction and (b) the overall tissue growth rate for segmented and mixed seeding distributions. Here, the ratio H is equal to 9.

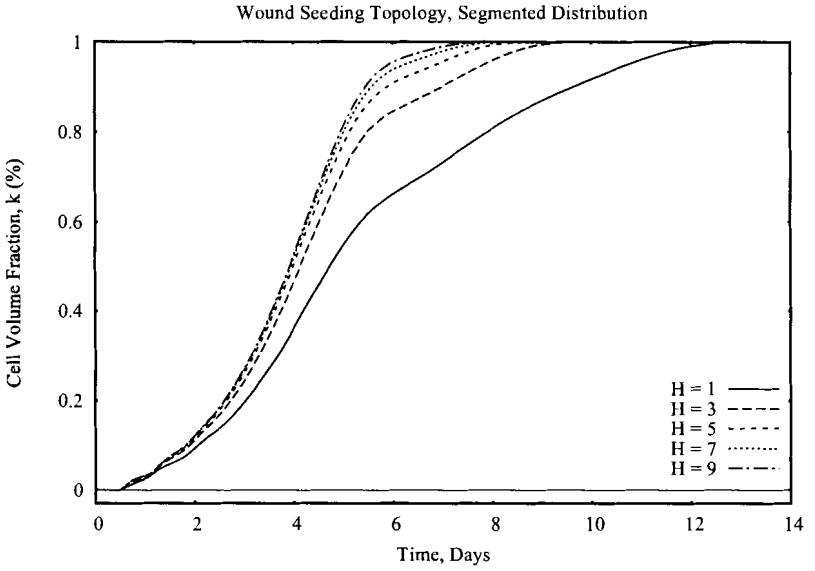
6.3.3 Wound Seeding Topology

Segmented Distribution

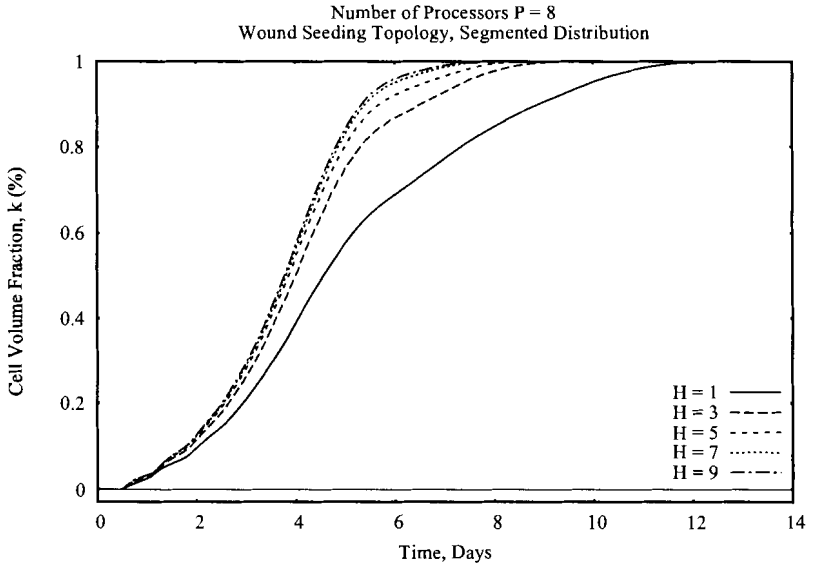
Effects of varying the ratio H

Figure 6.15 shows the effects of varying the ratio H on the cell volume fraction in a segmented wound-seeding distribution. In these simulations, cells in population 1 have a speed of 10 μm/hr while cells in population 2 have a speed of 1 μm/hr. As expected, when the value of H is increased, the time to reach full volume coverage is decreased. The corresponding growth rate curves when H is varied from 1 to 9 are shown in Figure 6.16 and Figure 6.17. We observe that higher values of H yield higher growth rates due to the fact that faster moving cells spread out in the denuded wound area preventing the formation of local clusters and hence proliferate faster.

Figure 6.18 and Figure 6.19 show the effects of varying the ratio H on the average number of collisions for populations 1 and 2, respectively. As time progresses, the average number of collisions increases initially and then decreases as cell colonies merge and confluence is reached. In Figure 6.18, as H decreases, there is a gradual decrease in the number of collisions. This is due in part to the fact that, for a lower value of H, the number of fast cells in population 1 proliferating into the empty cylinder is reduced. As time progresses, and due to their high speed, the cells proliferate into the cylinder and dominate the cell-to-cell interactions resulting in sustained collisions. In Figure 6.19, as H increases, the number of slow cells decreases, thus resulting in a smaller number of cell colonies being formed. This has a direct effect on reducing the average number of collisions for this cell population.

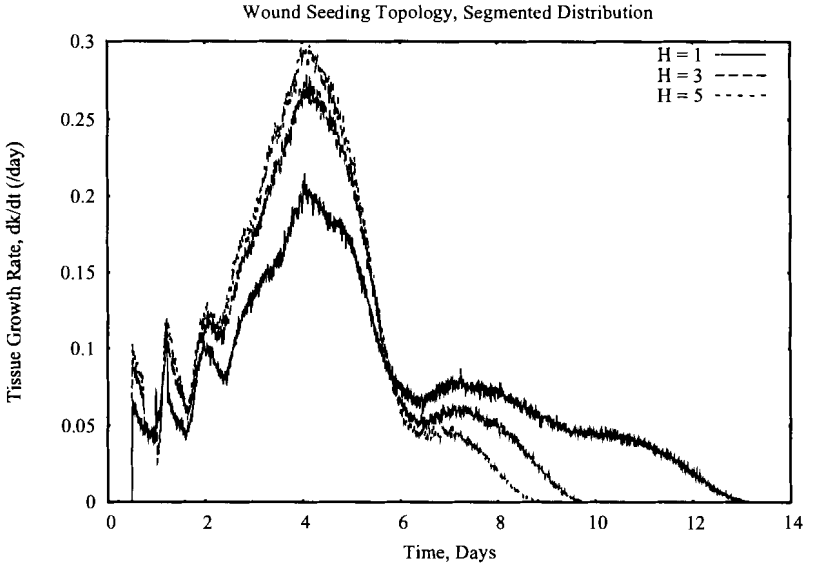


(a) Sequential Results

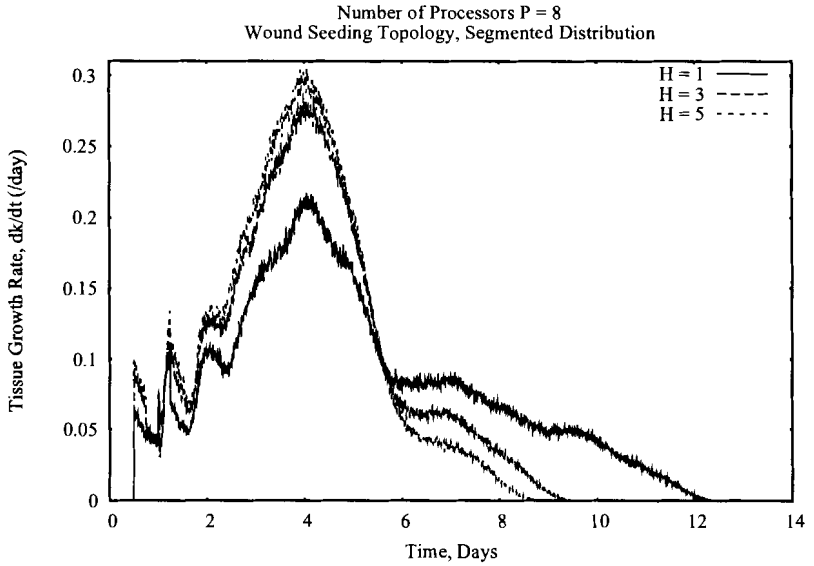


(b) Parallel Results

Figure 6.15: The effects of varying the ratio H on the cell volume fraction.

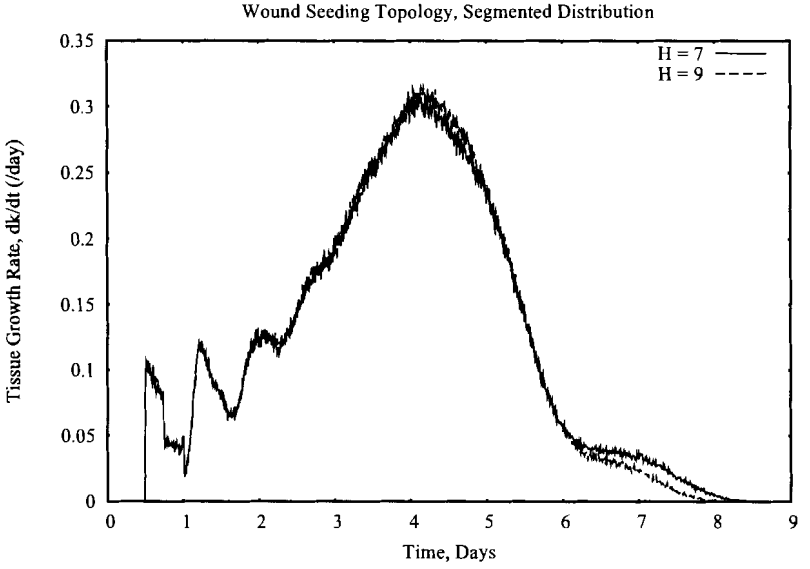


(a) Sequential Results

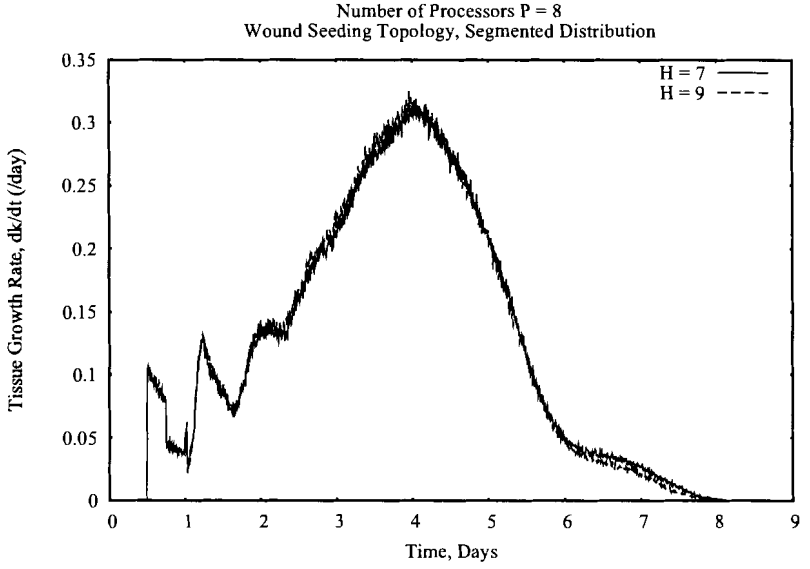


(b) Parallel Results

Figure 6.16: The overall tissue growth rate as the ratio H is varied from 1 to 5.

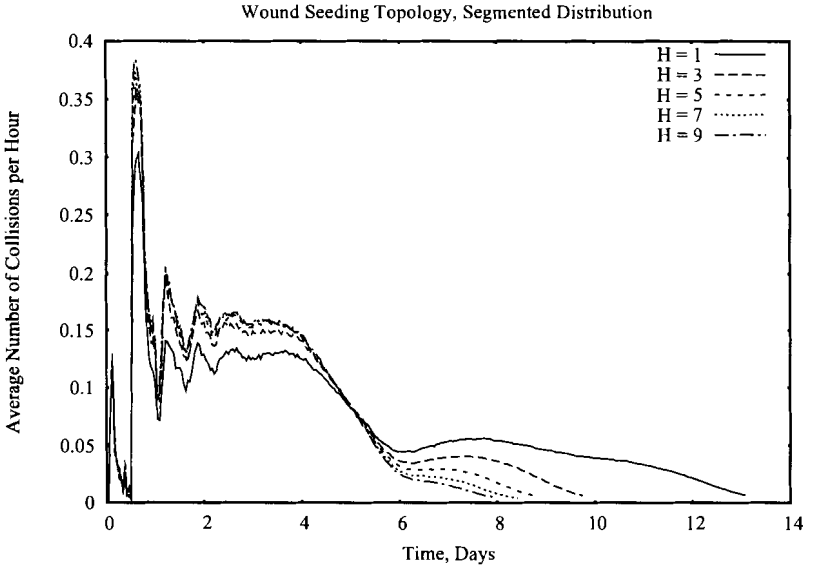


(a) Sequential Results

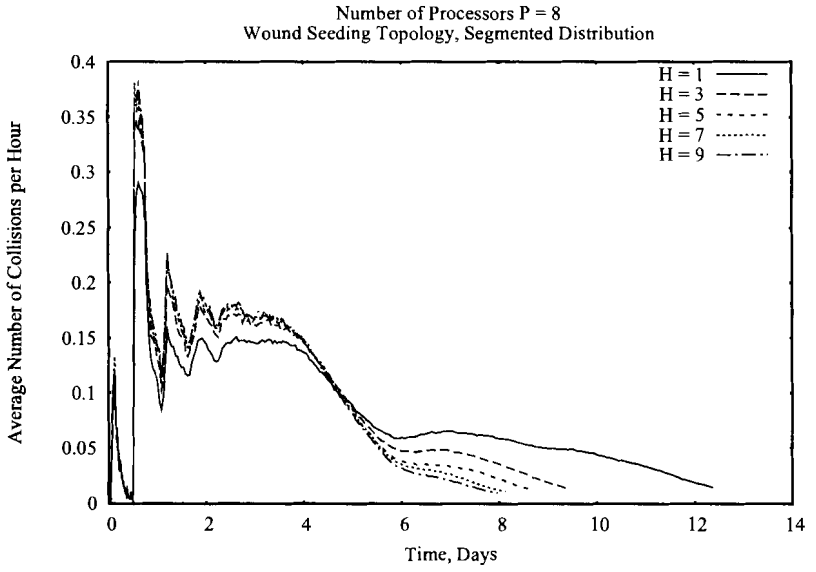


(b) Parallel Results

Figure 6.17: The overall tissue growth rate as the ratio H is varied from 7 to 9.

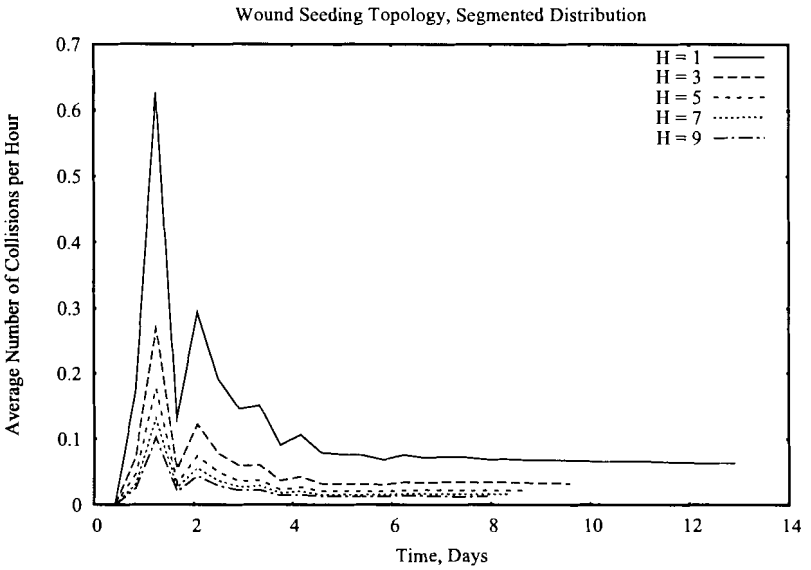


(a) Cell Population 1

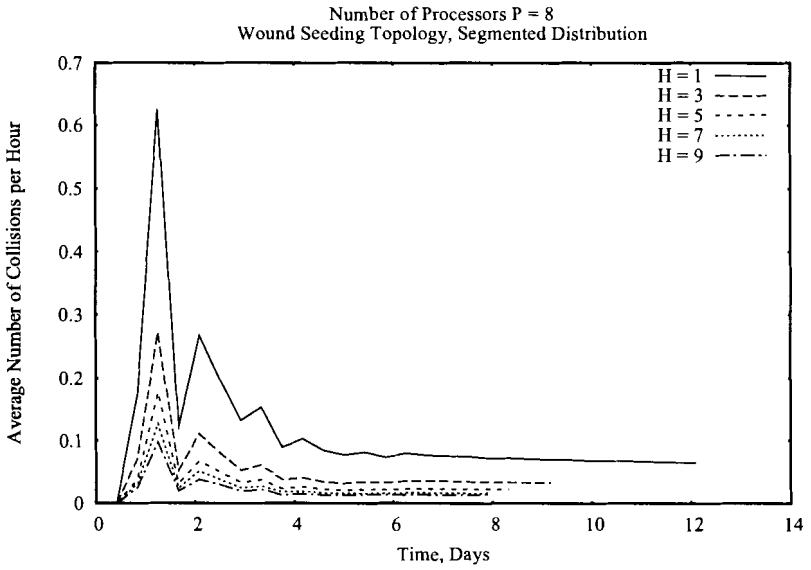


(b) Cell Population 2

Figure 6.18: The effects of varying the ratio H on the average number of collisions per hour for cell population 1 .



(a) Cell Population 1



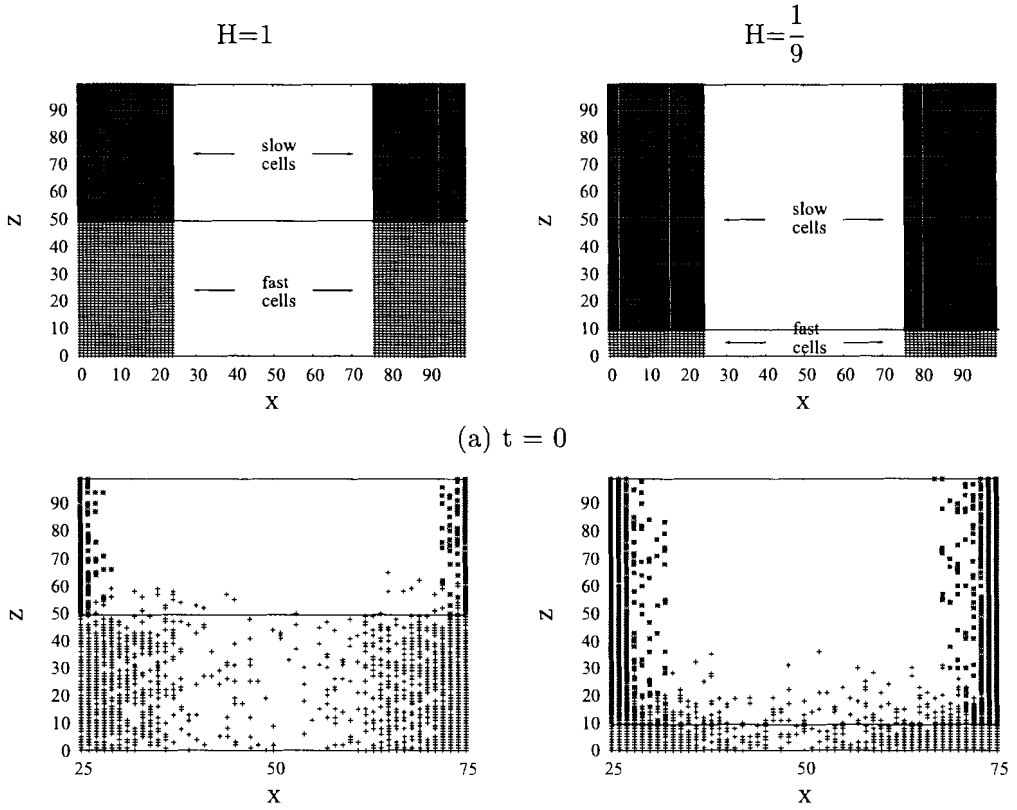
(b) Cell Population 2

Figure 6.19: The effects of varying the ratio H on the average number of collisions per hour for cell population 2.

Cell population profiles at different time steps

Figure 6.20 shows the cell population profiles of two simulation runs using a segmented wound seeding distribution and two different values of the ratio H ($H = 1$ and $\frac{1}{9}$, respectively). These profiles are exhibited as 2D cross sections of the wound area and at different levels of wound coverage. We observe the following:

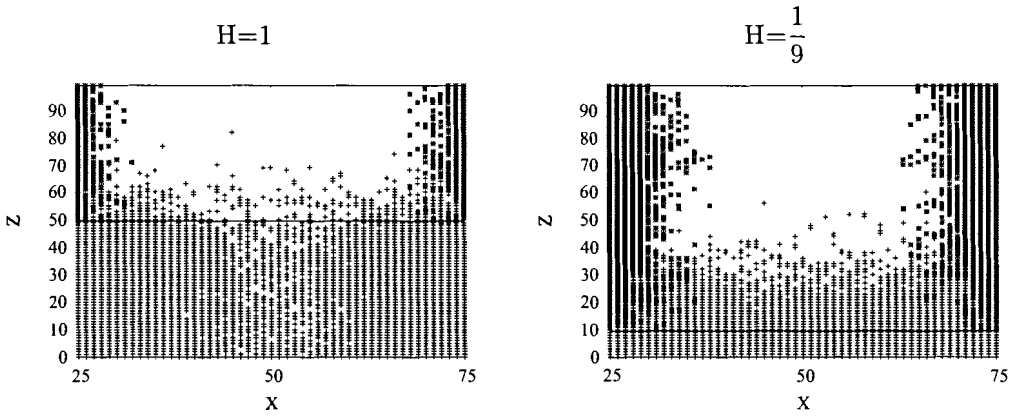
- At 33% of wound coverage, faster cells cover larger portions of the wound than slow cells as most of the growth appears in the lower partition of both seeding examples. There is also sporadic diffusion of fast cells into the upper partitions.
- At 66% of wound coverage, nearly most of the two lower partitions are covered while the upper portions remain mostly empty. In particular, when $H = \frac{1}{9}$, the lower partition is completely covered whereas the top segment is sparsely occupied by cells when $H = 1$.
- At 99% of wound coverage, the fast cells have penetrated to the top of the cellular space and have completely covered the denuded area at the bottom. Slow cells appear to mostly proliferate toward the center of the wound in the form of radial waves causing a cone-shaped funnelling of the diffusion of fast cells into the upper area of the wound.



(a) t = 0

(b) 33% of total coverage

Figure 6.20: Cell population profiles shown as 2D cross sections of two simulation runs at t = 0 and at 33% of wound coverage for $H = 1$ and $H = \frac{1}{9}$, respectively. For illustration purposes, we included a horizontal line to distinguish between the two cell populations (Part 1 of 2).



(c) 66% of total coverage

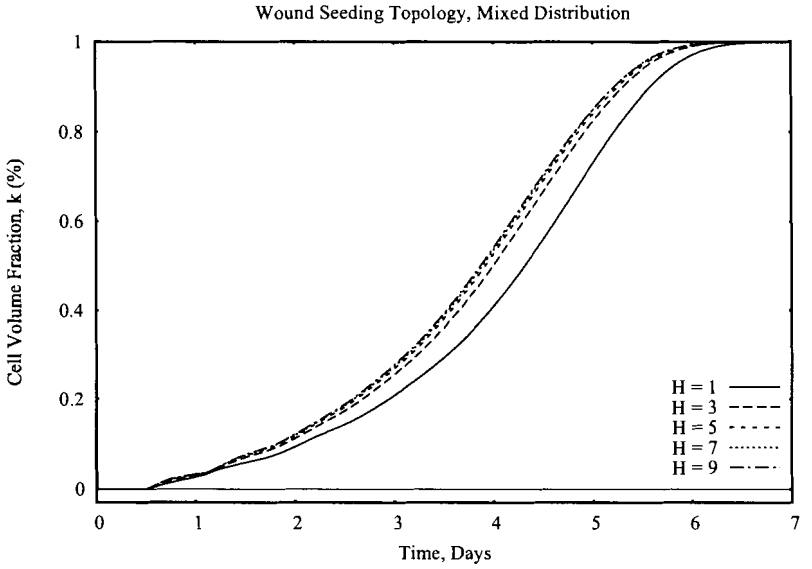
(d) 99% of total coverage

Figure 6.21: Cell population profiles shown as 2D cross sections of two simulation runs at 66% and 99% of wound coverage for $H = 1$ and $H = \frac{1}{9}$, respectively. For illustration purposes, we included a horizontal line to distinguish between the two cell populations (Part 2 of 2).

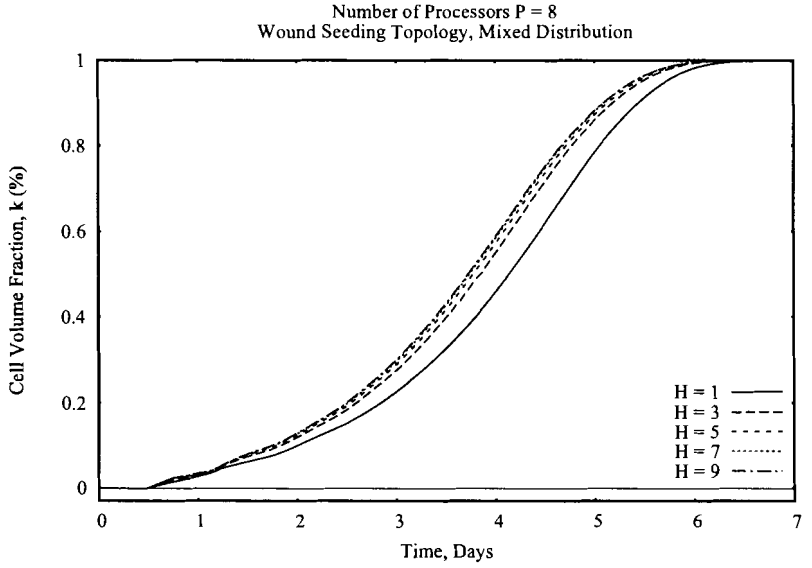
Mixed Distribution

Effects of varying the ratio H

Figure 6.22 shows the effects of varying the ratio H on the cell volume fraction in a mixed wound seeding distribution. In these simulations, cells in population 1 have a speed of 10 $\mu m/hr$ while cells in population 2 have a speed of 1 $\mu m/hr$. As the ratio H is increased, the time taken to reach confluence decreases. For values of $H > 5$, the relative impact of further increases becomes less discernible. This is also confirmed in Figure 6.23 and Figure 6.24 where the temporal evolution of the overall tissue growth rate is depicted for different values of H. Figure 6.25 and Figure 6.26 display the average number of collisions per hour versus time for cell population 1 and 2, respectively. Increasing the ratio H reduces the number of slow moving cells around the periphery of the wound which decreases the number of cell collisions as a result of fewer cell colonies formed. Faster moving cells take advantage of this setting by passing the slower cells and invading most of the wound area unobstructed.

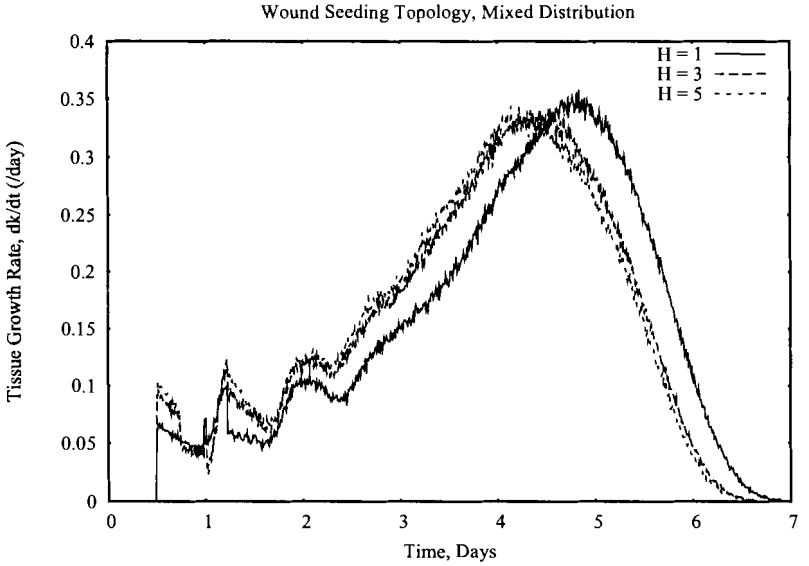


(a) Sequential Results

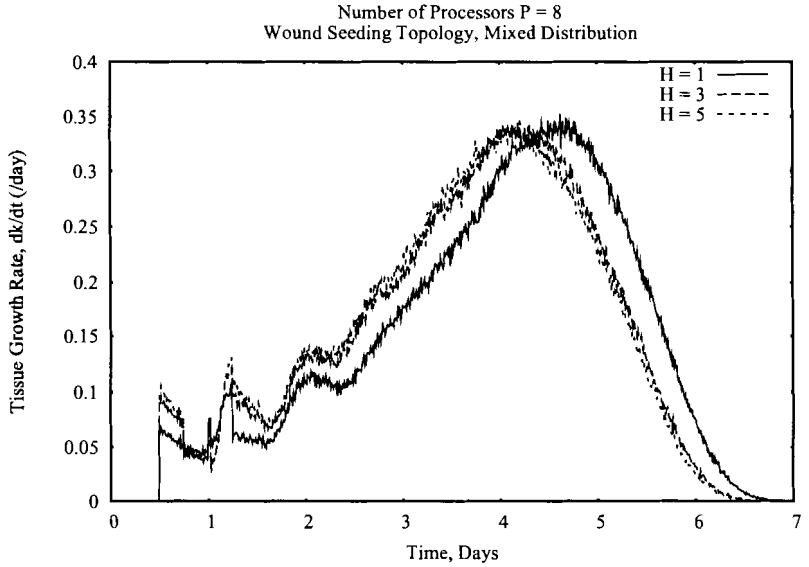


(b) Parallel Results

Figure 6.22: The effects of varying the ratio H on the cell volume fraction.

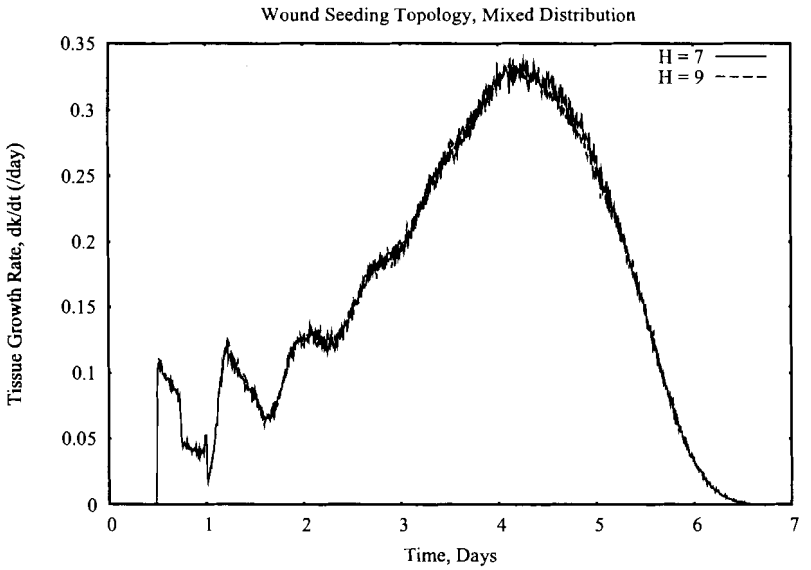


(a) Sequential Results

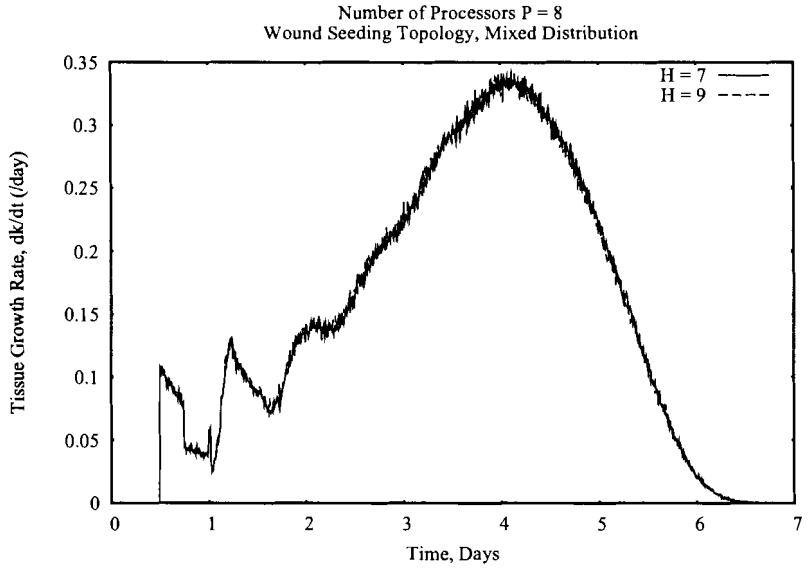


(b) Parallel Results

Figure 6.23: The overall tissue growth rate as the ratio H is varied from 1 to 5.

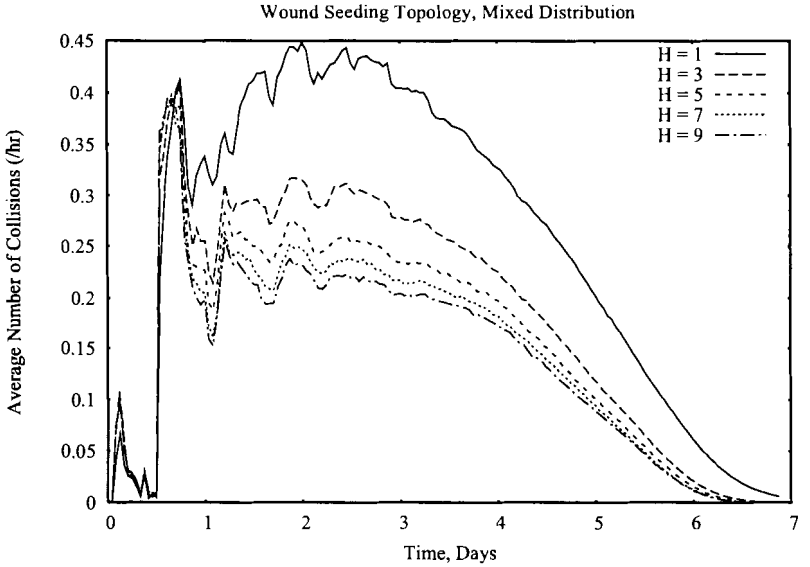


(a) Sequential Results

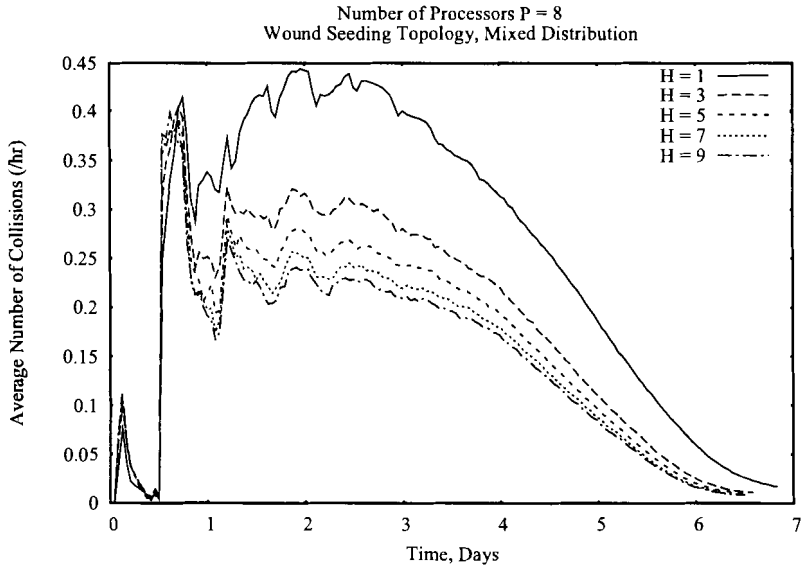


(b) Parallel Results

Figure 6.24: The overall tissue growth rate as the ratio H is varied from 7 to 9.

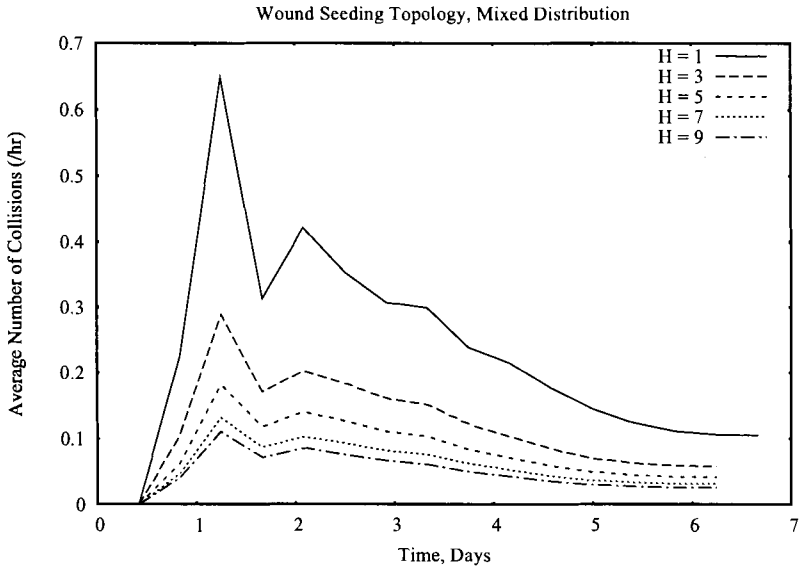


(a) Sequential Results

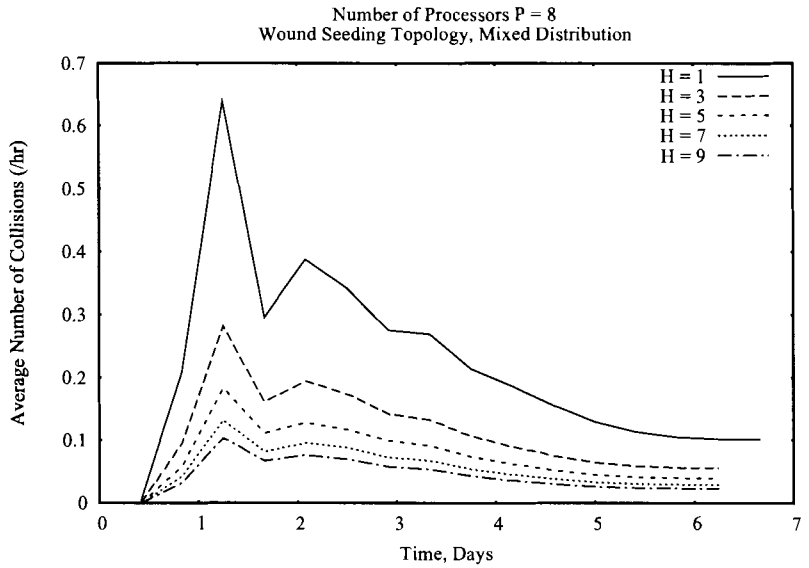


(b) Parallel Results

Figure 6.25: The effects of varying the ratio H on the average number of collisions per hour for cell population 1.



(a) Sequential Results

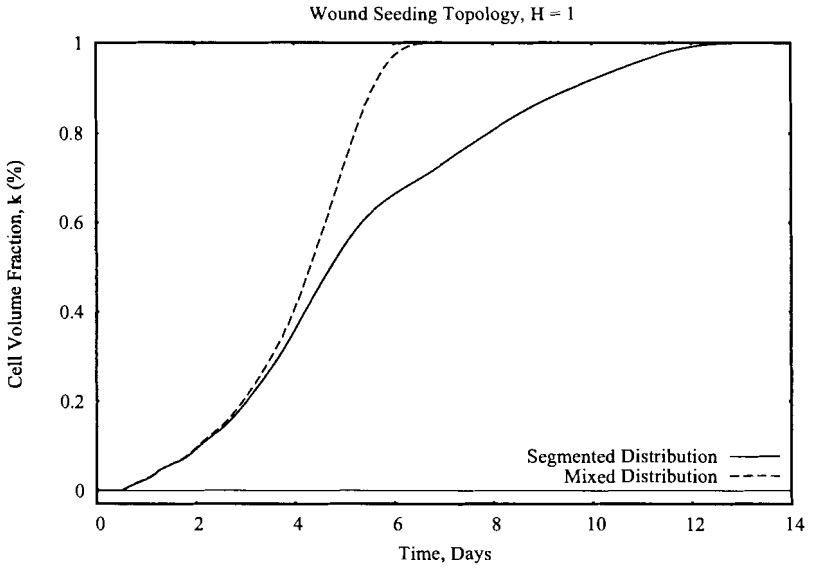


(b) Parallel Results

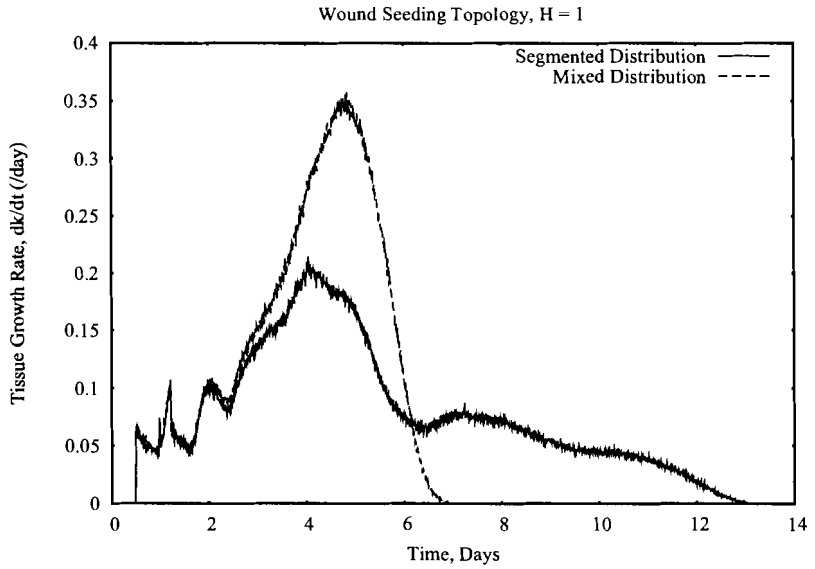
Figure 6.26: The effects of varying the ratio H on the average number of collisions per hour for cell population 2.

6.3.4 Comparison of Wound Seeding Cell Distributions

Figures 6.27 and 6.28 show comparisons between the cell volume fractions and the tissue growth rates obtained by using the segmented and mixed wound seeding distributions. Two different values of the ratio H were used, $H = 1$ and $H = 9$. On the other hand, the migration speeds of cells in population 1 and population 2 were kept constant at 10 $\mu m/hr$ and 1 $\mu m/hr$, respectively. When $H = 1$, the mixed distribution yields a faster proliferation rate and a higher tissue growth rate. Increasing H further to a value of 9 produces a stronger positive impact on the proliferation and tissue growth rates for the segmented distribution than the mixed one.

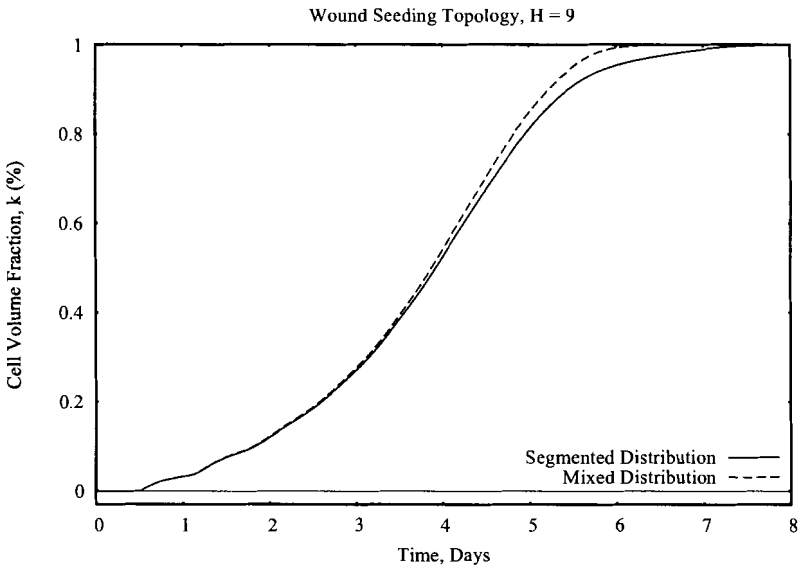


(a)

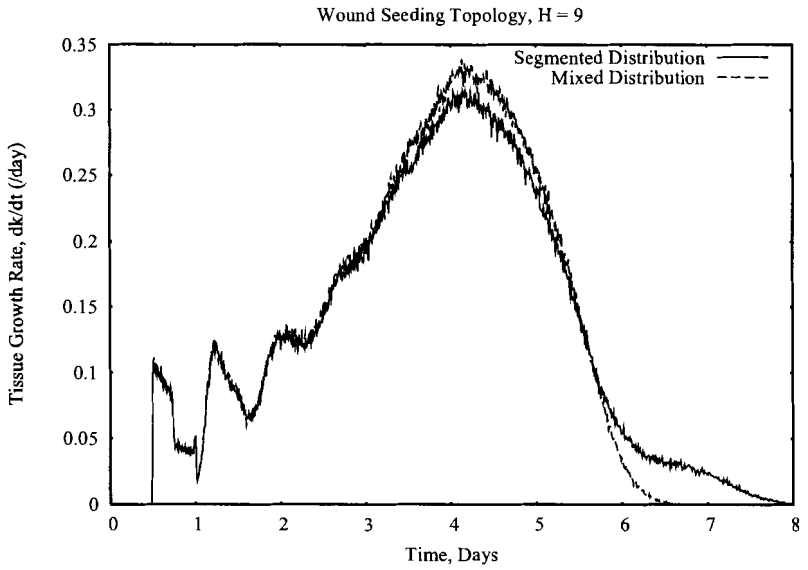


(b)

Figure 6.27: Comparison of (a) the cell volume fraction and (b) the overall tissue growth rate for segmented and mixed seeding distributions. Here, the ratio H is equal to 1.



(a)



(b)

Figure 6.28: Comparison of (a) the cell volume fraction and (b) the overall tissue growth rate for segmented and mixed seeding distributions. Here, the ratio H is equal to 9.

6.4 Chapter Summary

In this chapter, we presented our serial and parallel simulation results. These results were obtained by studying the effects of different seeding distributions, initial volume coverage, cell speeds, and cell heterogeneity on the proliferation and tissue growth rates of mammalian cells. In particular, we applied the slab decomposition technique for both the uniform and wound seeding topologies to obtain the parallel simulation results. The results from the parallel algorithm were found to be close to the results of the serial algorithm.

For both uniform and wound healing topologies, our simulations showed that increasing the ratio H of fast to slow cells enhanced tissue growth rates. For lower values of H ($H < 5$), the mixed cell distribution for both seeding topologies provided better results than the segmented one. As this ratio of cell heterogeneity increased beyond $H \approx 5$, the segmented cell distribution benefited far more than the mixed one in terms of higher tissue growth rates and faster times to reach confluence. This conclusion has significant implications for the design of experiments that can test the efficacy of cell heterogeneity and cell seeding distributions designed to enhance the overall tissue growth rate. To achieve these goals, it is recommended that assays based on these factors be adopted.

Chapter 7

Performance Analysis

We study the speedup and efficiency of our implementation of the slab decomposition technique for different processor numbers and cellular array sizes. The performance results were obtained by running both the sequential and parallel implementations on the Nebula Beowulf Cluster.

7.1 Measurement Conditions and Metrics

The following testing conditions were observed on the Nebula Beowulf cluster in order to generate consistent timing results in our simulation experiments:

- **Use the best available resources.** While access to the Nebula Beowulf cluster is allowed 24 hours a day, particular care was taken to run our simulations when the load on the cluster is the smallest (most experiments were run between 2:00 am and 8:00 am). The Portable Batch System (PBS) server handles the job management by allocating nodes to the various jobs, thereby freeing the user from processor management issues.

- **Use the best complier options.** We compiled our programs using

  mpic++ -O2 -march=pentium4 program.cpp – for the parallel program,

  and

  g++ -O2 -march=pentium4 program.cpp – for the sequential program.

The -O2 option provides the highest optimization level in the $g++$ complier without introducing errors into the application. The -march=pentium4 option instructs the compiler to generate Intel Pentium 4 processor-efficient code. Without it, the compiler would generate code with the generic PentiumPro instruction set, common to all i386 family chips.

- **Use high-level, portable C++ code.** Except for the standard I/O, timer libraries, and MPI, no other libraries or assembly code were used. We also did not rely on the system libraries to generate random numbers. Instead, we implemented our own random number generator, both sequentially and in parallel (see Chapter 4 and Chapter 5, respectively).

- **Measure both the execution and communication times.** We measured the execution time of the sequential program using the `clock()` function, which we monitor at a resolution of one microsecond. For parallel programs, we measure both the CPU and communication times using the MPI function `MPI_Wtime()`. Each program was executed at least four times (and at most eight times), and the best time result was used in our analysis, because it corresponds to the simulation run experiencing the least interference from the operating system.

We define below some performance metrics used in our work. The terminology is consistent with that provided in the parallel processing literature ([20] and [51]). For our program, we applied the following performance metrics:

Definition 9.1 *The Sequential Execution Time, T(1), is the fastest total execution time (in seconds) of the sequential program running on one node.*

Definition 9.2 *The Parallel Execution Time, T(P), is the fastest total execution time (in seconds) of a parallel program running on P nodes, including communication time.*

Definition 9.3 *The Speedup, S, and Efficiency, E, are defined by the ratios* $S = \frac{T(1)}{T(P)}$ *and* $E = \frac{T(1)}{P \times T(P)} = \frac{S}{P}$, *respectively.*

Definition 9.4 *The Total Communication Time,* T_{comm} *(P), is the total time spent by a parallel program running on P nodes performing communication operations.*

There are three types of communication operations on the cluster: point-to-point communication, collective communication, and aggregated computation. In a *point-to-point communication*, only two nodes, the sender and the receiver, are involved. In a *collective communication* operation, tasks in a group send messages to one another, and the time is a function of both the message length and the number of nodes. In a *broadcast* operation, a single node sends an m-byte message to the remaining $P-1$ nodes. Other examples include *gather* and *scatter* operations. In an *aggregated computation*, tasks in a group synchronize with one another or aggregate partial results. The time for such an operation is a function of the group size, but not of the message length, as the message length is fixed. For example, in a *barrier* operation, a group of tasks synchronize with one another, that is, they wait until all tasks execute their respective barrier operation. In a *reduction* operation, a group of tasks aggregate partial results, one from each task, into a final result. Examples include finding the maximum of P values and performing a parallel *prefix* or *scan* operation.

7.2 Performance Results and Discussion

The time complexity of our implementation of the sequential algorithm is of the order of $O(N^3)$. In addition, our implementation of the parallel algorithm based on the slab decomposition has the time complexity of the order of $O(\frac{N^3}{P})$. Speedup and efficiency are two of the most commonly accepted performance measurements of an application running on a parallel computer system. The sequential execution time is obtained by running the serial algorithm on a single node of the cluster.

We were limited by the available memory capacity per node on the cluster. For instance, the largest cellular array size for the sequential runs was $330 \times 330 \times 330$. The following performance results were obtained for a uniform segmented distribution with an initial seeding density of 0.5%, a ratio $H = 1$, and cell migration speeds of 10 $\mu m/hr$ and 1 $\mu m/hr$ for cell population 1 and cell population 2, respectively. We note that similar performance results were obtained by the other three types of cell seeding distributions.

For each cellular array size, we vary the number of processors P from 2 to 25 and for each selected number of processors in this range, we vary the size of the cellular array such that $N \in \{150, 200, 250, 300, 330\}$. The execution time, speedup, efficiency, and the communication time for different cellular array sizes and numbers of processors are shown in Tables 7.1 to 7.4. Moreover, Figures 7.1 to 7.4 show a comparison of the values of

the speedup and efficiency for various cellular array sizes and numbers of processors. The performance results show that for a given number of processors and as the cellular array size increases, the speedup and efficiency values increase as well. This is due to the fact that as the size of the cellular array increases, the ratio of the execution time over the communication time of each node increases as clearly indicated by Table 7.2 and Figure 7.6.

In addition, the performance results show that for a fixed cellular array size and as the number of processors increases, the speedup value increases while efficiency decreases. The increase in speedup is due to the fact that increasing the number of processors yields, for most cases, smaller execution times. On the other hand, the decrease in efficiency is due to the fact that increasing the number of processors increases the communication time for a given problem size (N) and, thus increases its related overhead. These two effects are depicted in Figure 7.5. This overhead may comprise idling, communication, and excess computation. It is interesting to note that for the cellular array of $150 \times 150 \times 150$, the speedup reaches its maximum value of approximately 3.94 when P is in the range from 10 to 20 processors and then starts to decrease for larger values of P as a consequence of *Amdahl's law*. Here, we observe that the communication time dominates the parallel execution time and accounts for 83% of it as shown in Table 7.2. As a result, the parallel execution time at $P = 25$ starts to increase and is noticeably greater than the parallel execution time at $P = 20$ as illustrated in Table 7.1. For $N = 150$, the parallel algorithm has a performance sweet spot defined by $P \in [10, 20]$.

Table 7.1: Execution times in seconds for various cellular array sizes and different numbers of processors.

Cellular Array	**Number of Processors**						
	1	**2**	**4**	**8**	**10**	**20**	**25**
$150 \times 150 \times 150$	2124	1441.6	861.0	570.5	538.9	539.2	571.1
$200 \times 200 \times 200$	5413	3568.6	1942.2	1181.0	1047.2	838.8	816.8
$250 \times 250 \times 250$	11382	7389.2	3911.5	2292.2	1925.3	1361.5	1255.0
$300 \times 300 \times 300$	21211	13199.7	6884.0	3838.3	3291.8	2029.9	1833.5
$330 \times 330 \times 330$	29156	17873.0	9255.1	5072.3	4266.5	2615.0	2298.9

Table 7.2: Communication times in seconds (and percent communication) for various cellular array sizes and different numbers of processors. Percent communication is the percentage of the parallel execution time spent in communications.

Cellular Array	Number of Processors					
	2	**4**	**8**	**10**	**20**	**25**
150 × 150 × 150	75.9 (5%)	152.6 (18%)	242.8 (43%)	295.5 (**55%**)	414.9 (77%)	475.7 (83%)
200 × 200 × 200	98.3 (**3%**)	201.0 (**10%**)	324.4 (**27%**)	379.9 (**36%**)	527.2 (**63%**)	568.4 (**70%**)
250 × 250 × 250	136.2 (2%)	290.5 (7%)	418.4 (18%)	503.1 (**26%**)	687.3 (50%)	734.4 (59%)
300 × 300 × 300	146.0 (1%)	459.1 (7%)	571.6 (15%)	701.1 (**21%**)	825.4 (41%)	869.9 (47%)
330 × 330 × 330	178.8 (1%)	483.0 (5%)	675.6 (13%)	752.6 (**18%**)	915.7 (35%)	944.5 (41%)

Table 7.3: Speedup values for various cellular array sizes and different numbers of processors.

Cellular Array	Number of Processors						
	1	**2**	**4**	**8**	**10**	**20**	**25**
150 × 150 × 150	1.00	1.47	2.47	3.72	3.94	3.94	3.72
200 × 200 × 200	1.00	1.52	2.79	4.58	5.17	6.45	6.63
250 × 250 × 250	1.00	1.54	2.91	4.97	5.91	8.36	9.07
300 × 300 × 300	1.00	1.61	3.08	5.53	6.44	10.45	11.57
330 × 330 × 330	1.00	1.63	3.15	5.75	6.83	11.15	12.68

Table 7.4: Efficiency values for various cellular array sizes and different numbers of processors.

Cellular Array	Number of Processors						
	1	**2**	**4**	**8**	**10**	**20**	**25**
150 × 150 × 150	1.00	0.74	0.62	0.47	0.39	0.2	0.15
200 × 200 × 200	1.00	0.76	0.7	0.57	0.52	0.32	0.27
250 × 250 × 250	1.00	0.77	0.73	0.62	0.59	0.42	0.36
300 × 300 × 300	1.00	0.8	0.77	0.69	0.64	0.52	0.46
330 × 330 × 330	1.00	0.82	0.79	0.72	0.68	0.56	0.51

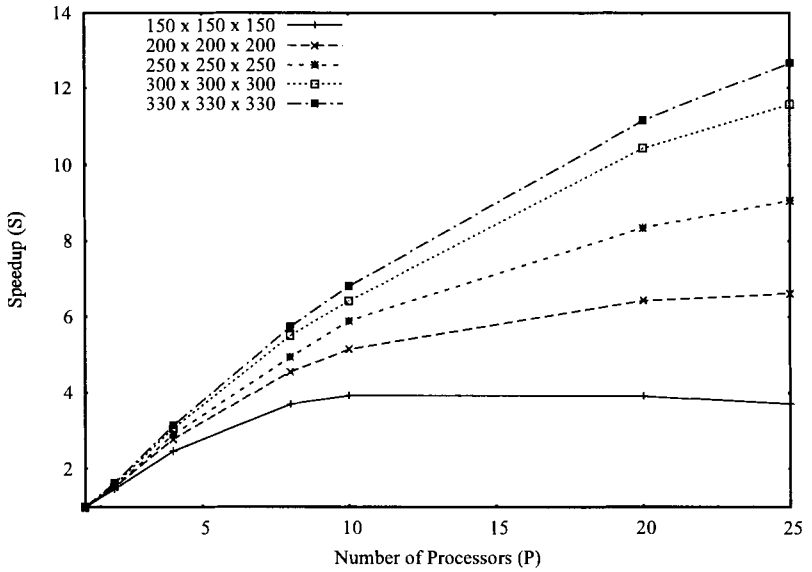


Figure 7.1: Comparison of speedup curves for various cellular array sizes. Each curve plots the speedup versus the number of processors for a given cellular array size.

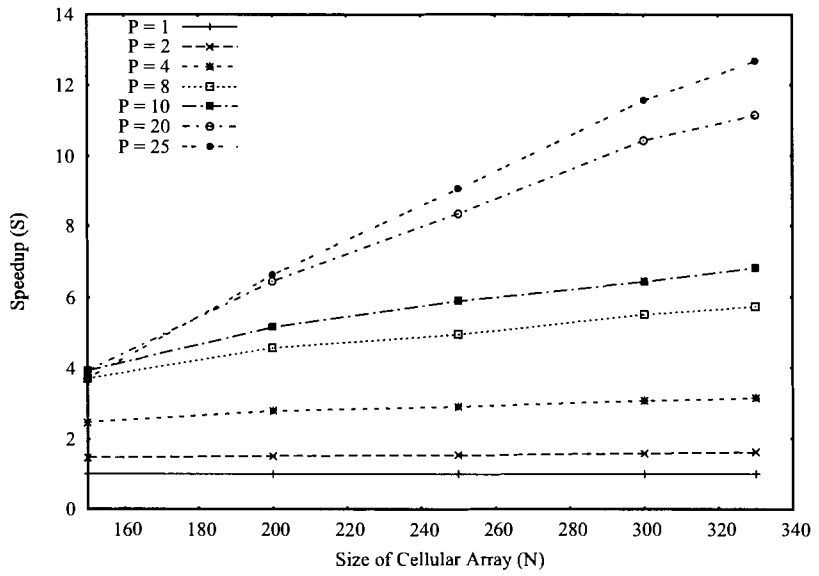


Figure 7.2: Comparison of speedup curves for various number of processors. Each curve plots the speedup versus the size of the cellular array for a fixed number of processors.

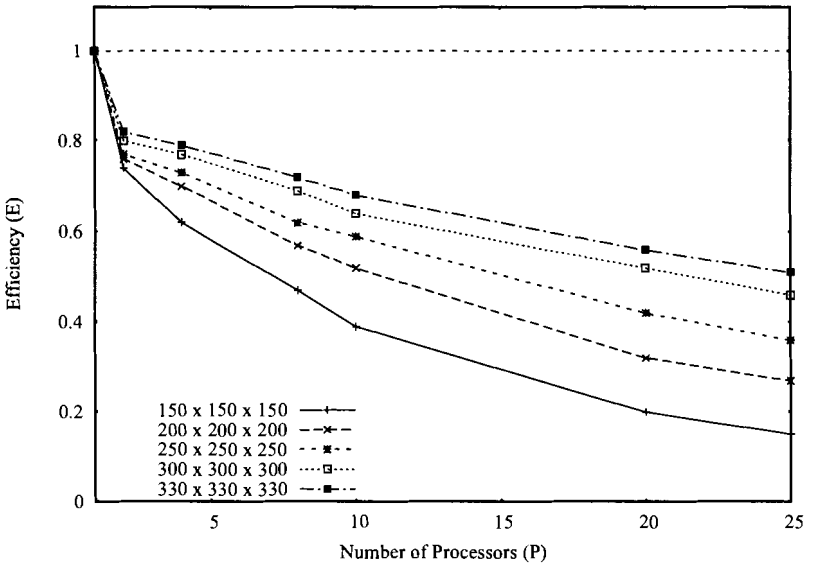


Figure 7.3: Comparison of efficiency curves for various cellular array sizes. Each curve plots the efficiency versus the number of processors for a given cellular array size.

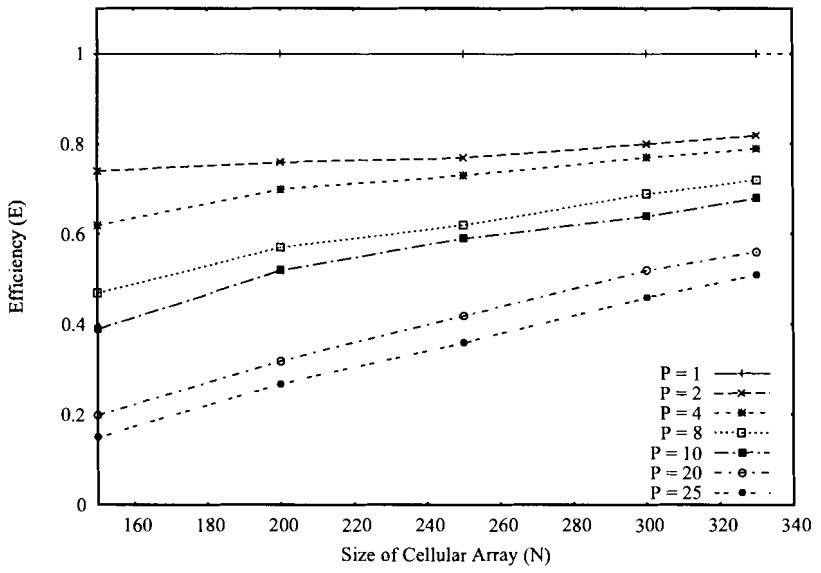


Figure 7.4: Comparison of efficiency curves for various number of processors. Each curve plots the efficiency versus the size of the cellular array for a fixed number of processors.

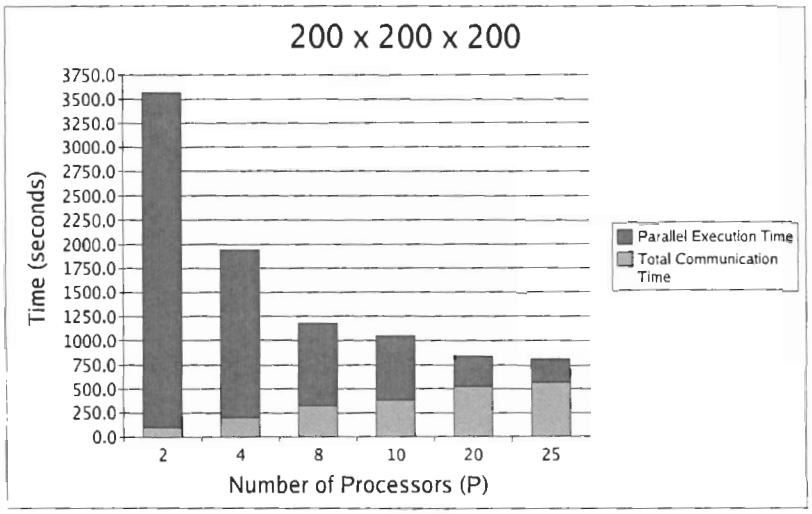


Figure 7.5: Plot of parallel execution time and total communication time versus number of processors (P) for a cellular array size of $200 \times 200 \times 200$.

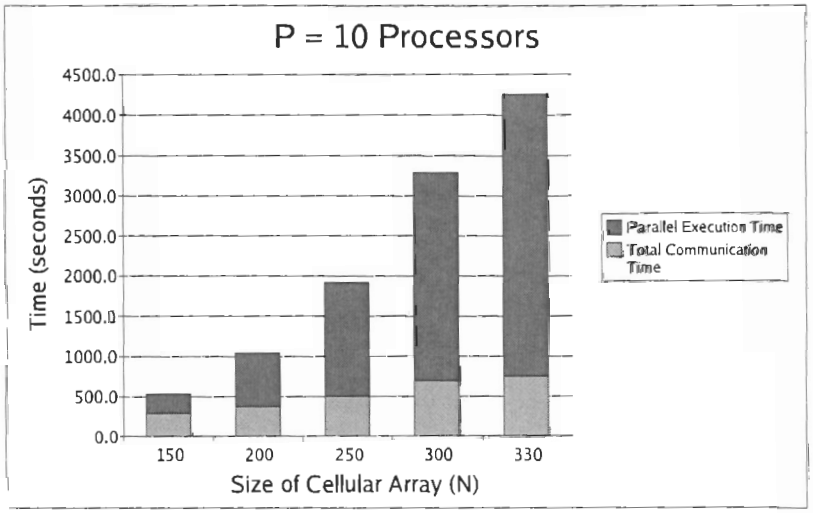


Figure 7.6: Plot of parallel execution time and total communication time versus cellular array size (N) for $P = 10$ processors.

The presented performance results in terms of speedup and efficiency are good especially for cellular array sizes of $200 \times 200 \times 200$ and larger (up to $N = 330$) as well as for numbers of processors between 2 and 25. Given these results, we make the following observations:

1. For a given cellular array size, as the number of processors is increased, we get a better speedup. By increasing the number of processors, each processor is required to process a smaller subdomain and the total execution time decreases resulting in an increased speedup.

2. For a given number of processors, as the cellular array size increases, we get a better speedup. By increasing the cellular array size, each processor is required to process a larger subdomain. The ratio of execution time over the communication time is increased resulting in a higher speedup value.

3. For a given number of processors, as the cellular array size increases, the efficiency increases. By increasing the cellular array size, there is an increased ratio of processing time to communication time. This leads to an increase in the time portion during which the processor is effectively employed, thus resulting in greater efficiency.

4. For a given cellular array size, as the number of processors is increased, the efficiency decreases. By increasing the number of processors, the communication time is increased due to an increase in overhead introduced with the addition of processors. This results in a decrease in the time during which the processors are effectively employed, thus yielding lower efficiency values.

7.3 Chapter Summary

We presented the speedup and efficiency results of our implementation of the slab decomposition method for different processor numbers and cellular array sizes. The obtained performance results were good especially for moderate and larger cellular array sizes. In particular, for a fixed number of nodes the slab decomposition scheme exhibited improved speedup and efficiency as we increased the size of the cellular array.

Chapter 8

Conclusion

8.1 Contributions

The main contribution of this research work is the extension of a previously developed three-dimensional model by Ben Youssef [3] and its parallel implementation on a distributed-memory Beowulf Cluster. The extended model incorporates multiple cell types and accounts for cell aggregation. The thesis presents the sequential and parallel results related to two populations of cells each with its own migratory and division characteristics.

8.1.1 The Three-Dimensional Model

The three-dimensional model uses a discrete approach based on cellular automata to study the tissue growth rates and population dynamics of migrating and proliferating mammalian cells. Cell migration is modelled using a discrete-time Markov chain approach. We have two populations of cells in the three-dimensional cellular space, each with its own division and motion characteristics. Both of these cell types have different division characteristics based on experimental data. The aggregation of cells of the same type was also accounted for.

The algorithm was implemented sequentially and then parallelized using the slab decomposition method. We were able to achieve our objective to predict the time required to reach complete volume coverage when we know the properties of mammalian cell locomotion and division. The motion and division characteristics of the two populations of cells are varied at the beginning of the simulation to study their effect on the tissue growth rate and population dynamics of cells.

The research will help speed up progress in the area of tissue engineering where the development of bio-artificial tissues involves extensive and time-consuming experimentation. It also extends the existing features of the model to incorporate features that more realistically simulate the process of cell migration and proliferation.

8.2 Future Work

The growth of three-dimensional tissues with specific cellular structure and function is a complicated process affected by many physical and biological parameters. The morphology (pore structure, roughness, connectivity of pores, etc.) and surface composition of the scaffolds affect cell migration and proliferation. At the same time, cell growth and function is affected by external stimuli like growth factors or cytokines and by the availability of specific nutrients. The present research can be extended to more complex models that take into account cell death and nutrient limitations. Other work will involve modifying the tissue growth model to simulate tumour growth and integrating with the current model and succeeding ones a corresponding visualization solution.

8.2.1 Cell Death

Previous models, including the one presented in this thesis, accounted for the motion and proliferation of cells but did not take into consideration the death of cells, also known as *apoptosis* (one of the main types of programmed cell death). By taking into account the death of cells, we can more realistically model the proliferation of cells, tissue growth, and the time to reach confluence. The death of cells may be incorporated into the model as follows: cells die after a certain number of divisions dependent on an initial generation number and a death counter both of which can be randomly selected from normal distribution data. When a cell dies it stays in its position in the cellular space for a fixed amount of time, called decomposition time, after which the cell decomposes and the site it occupies is free for a neighbouring cell to migrate to.

8.2.2 Nutrient Limitations

Due to the diffusional limitations of nutrients in a scaffold, the maximum thickness of tissues grown in non-vascularized scaffolds does not exceed 0.1 to 0.2 millimetres. Our ability to

grow large tissues or organs will depend on whether we can increase nutrient availability in the interior of the scaffold either by flowing media through it or by pre-vascularizing the scaffold.

Simulations can be used to assess the effect of these two approaches. Using known mass transfer correlations, we can solve the problem of diffusing nutrients with and without fluid flow through the scaffold. The cell growth models will then be adjusted to account for nutrient limitations, i.e., the models will assume that cells migrate and proliferate only where the nutrient concentration is above a certain threshold. Since the cells consume nutrients, there will be a maximum cell density that can be supported for each nutrient concentration level.

We will also consider the case where blood vessels start forming at the periphery of an implanted scaffold and move inside, bringing more nutrients to the interior of the scaffold. Thus, the nutrient concentration profiles become much more complicated and can only be treated numerically. The cellular automata approach is ideally suited to treat such problems with complicated geometry.

8.2.3 Tumor Growth

Most cancers derive from a single abnormal cell that has undergone a change in its DNA (DeoxyriboNucleic Acid) sequence via mutation [1]. If the mutation enables cells to proliferate in defiance of normal restraints and invade and colonize territories normally reserved for other cells, a cancer results. Depending on whether or not they can invade the surrounding tissue, tumors are defined as invasive (malignant) or noninvasive (benign). The progression of invasive cancer is often categorized as Stage I, II, III, or IV [2]. Stages I and II are considered the early stages in the sense that the tumor has not yet spread to distant parts of the body through the blood stream or lymphatic vessels to form secondary tumors (metastasis). Most mathematical modeling of tumor growth focuses on this stage because that is when the tumor is still well defined and treatments are likely to be most effective.

In order to invade the surrounding tissue, cancer cells must have competitive phenotypes enabling them to outgrow, or suppress the growth of, normal cells. In addition, it has been also presented that cancer cells preferentially convert nutrients (such as glucose) to lactic acid. This phenotype results in a several fold increase in glucose uptake and lactic acid production in tumor cells versus normal cells. In fact, the increase is so marked it is now routinely used to diagnose many types of tumors and to monitor early tumor response to

therapies [27]. As normal cells die and decompose, tumor cells can proliferate and take over their position in the tissue (tumor invasion). This mechanism of acid-mediated tumor growth (AMTG) has been supported by many recent reports that high lactate levels are directly correlated with the likelihood of metastases, tumor recurrence and restricted patient survival [60].

We believe that a quantitative understanding of AMTG can help scientists better understand the effects of the aforementioned key factors on tumor growth and, thereby, design more effective cancer treatment medicines/strategies. The complexity of the process necessitates a systematic approach similar to the one we undertook to model tissue growth. The help of predictive computational models is even more important here because clinical trials in this area usually involve human subjects and are thus much more expensive and time-consuming than in vitro tissue growth.

8.2.4 Visualization

An important future task includes extending the real-time visualization of tissue growth to support multiple types of cells. As we delve into the visualization of multi-cellular tissues, a new set of challenges is presented to the visualization process. The use of direct, three-dimensional, volume rendering algorithms becomes necessary in order to be able to show the internal structures as well as the global features of the newly grown tissue [8]. Moreover, there is currently a large number of tunable parameters for the simulation model. Therefore, the use of high-level parameter abstraction to reduce the size of the parameter space would enhance the usability of the visualization tools planned for development. Other supporting features for the design space exploration that can be added include undo, redo, and fast tissue growth preview capabilities.

8.2.5 Other Parallel Architectures

In Chapter 7, we showed that, in the case of our parallel model, the communication time is a significant bottleneck in the performance of the parallel algorithm. Using a shared-memory architecture may yield faster overall execution times. In a shared-memory multiprocessor, the architecture is characterized by a single global memory that is shared by multiple nodes. This is enabled by a fast interconnection network. For a multi-threaded application, each thread is executed on a dedicated processor. Since the threads are running asynchronously,

care must be taken to synchronize the processors at each time step. Moreover, mutual exclusion may be required to access critical regions of memory where data are shared. Shared-memory systems are complicated by the fact that multiple processors may attempt to access the same memory address. To maintain correctness, a node will be required to mark the cell that is being processed along with the six neighbouring sites. Once processed, the cell and its six neighbouring sites can be released and can be accessed by other processors. This will prevent multiple processors from simultaneously writing to the same computational site. There are two observations when using a shared-memory architecture. First, the communication overhead is anticipated to be low. Second, the communication performance may only be limited by the bandwidth of the interconneciton network.

Appendix A

Additional Simulation Results

In the following sections of this appendix, we present additional serial and parallel simulation results. In these simulations, we compute the tissue growth rates for various initial seeding densities and cell population 1 speeds for the mixed uniform cell distribution. Next, the cell speeds of cell population 1 and cell population 2 are varied to compare their effects for both the segmented and mixed uniform distributions. In the last set of results, the effects of varying cell population 1 speeds are observed in the wound-seeding topology using a mixed distribution. As previously noted, all the parallel results are similar to the serial ones and therefore the same discussion applies.

A.1 Uniform Seeding Topology

A.1.1 Mixed Distribution

A.1.2 Effects of varying seeding density

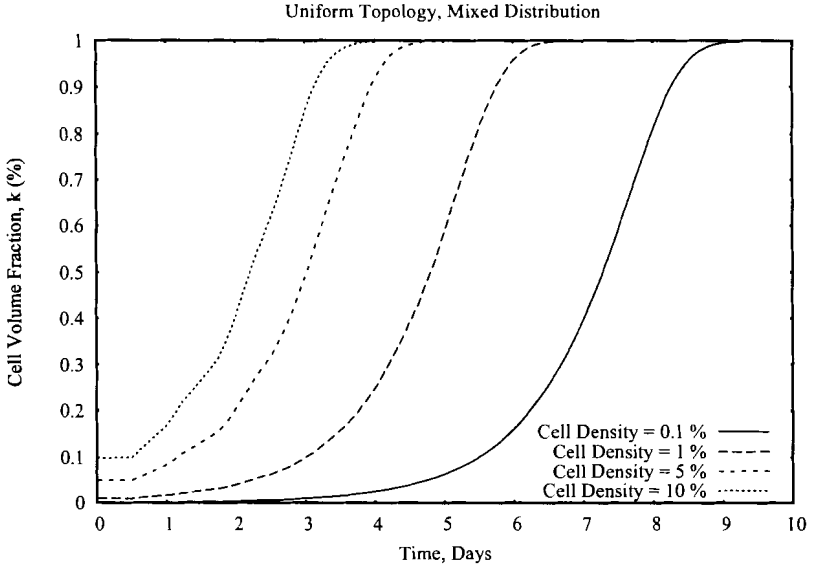
Figure A.1 and Figure A.2 show the effects of varying the seeding density on the cell volume fraction and the tissue growth rate, respectively. Here, cells from both populations are seeded in a mixed distribution using a uniform topology with a ratio H equal to 1. The total seeding density is varied from 0.1% to 10%. That is, the initial seeding density of both cell populations has values that varied from 0.05% to 5%.

As expected, the time required to reach confluence decreases with increasing the initial seeding density. For very low initial volume coverage, the curves have a very long induction period because of slow growth of isolated colonies dominating the process. As the initial

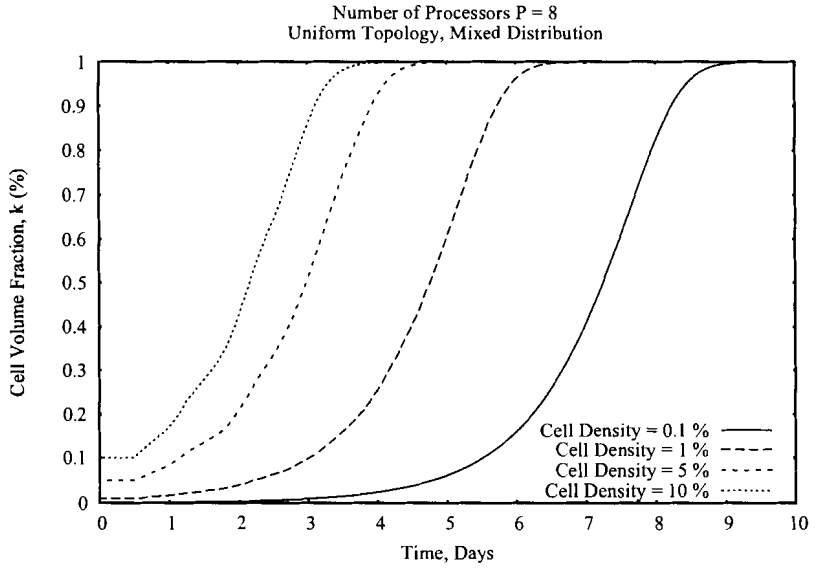
volume coverage increases, this induction period disappears. For high seeding densities, the volume coverage and the tissue growth rate increase rapidly as the cells reach confluence within a few divisions and before isolated colonies can form.

For higher seeding densities, the impact of cell division can be readily observed by the presence of multiple peaks on the corresponding growth rate curve. Figure A.3 illustrates this phenomenon in the case of a seeding density of 10%. According to the division time distributions of population 1 and population 2, 64% and 4% of cells, respectively, will divide between 12-18 hours. This is depicted by the first plateau on the three shown curves. Additional plateaus are observed at later times reflecting the succeeding waves of cell divisions occurring during the proliferation process. The subsequent waves of cell divisions include not only seeding cells giving birth to new cells but also daughter cells going through their own mitotic cycles.

Figure A.4 and Figure A.5 show the effects of varying the seeding density on the average number of collisions for the two populations of cells. There is a small number of collisions at the beginning of the simulations due to the fact that cells are initially seeded in a random and sparse fashion. Then, as the volume coverage increases, the average number of collisions also increases due to an increase in cell-cell interactions. Shortly after reaching 75–80% of confluence, the average number of collisions starts to decrease rapidly due to the merging of small cell colonies. As the initial seeding density increases, the average number of collisions for both cell populations increases while the peak value and the drop to zero occur sooner.

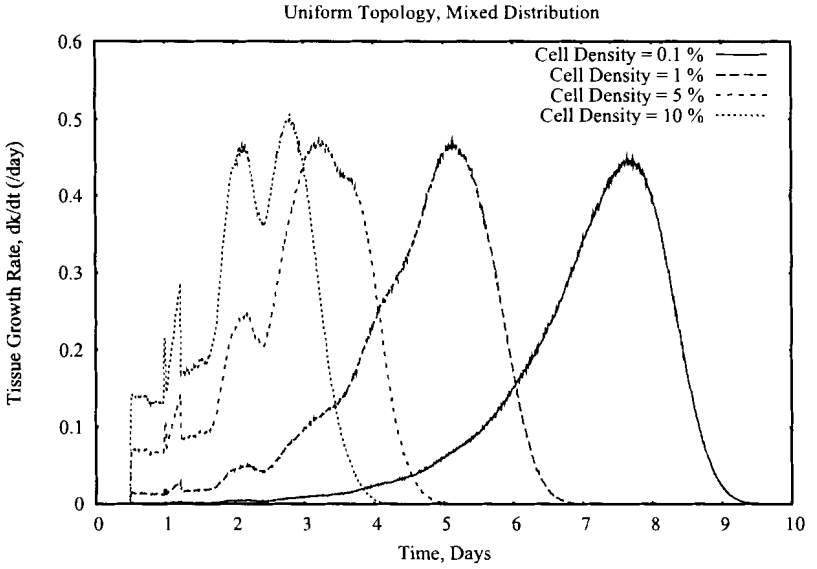


(a) Sequential Results

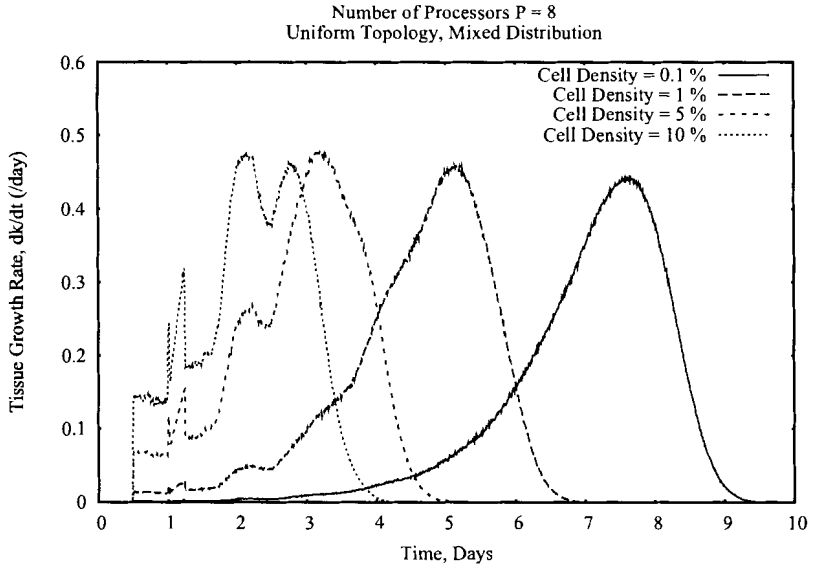


(b) Parallel Results

Figure A.1: The effects of varying the total cell seeding density on the cell volume fraction. The ratio H is held constant at 1.

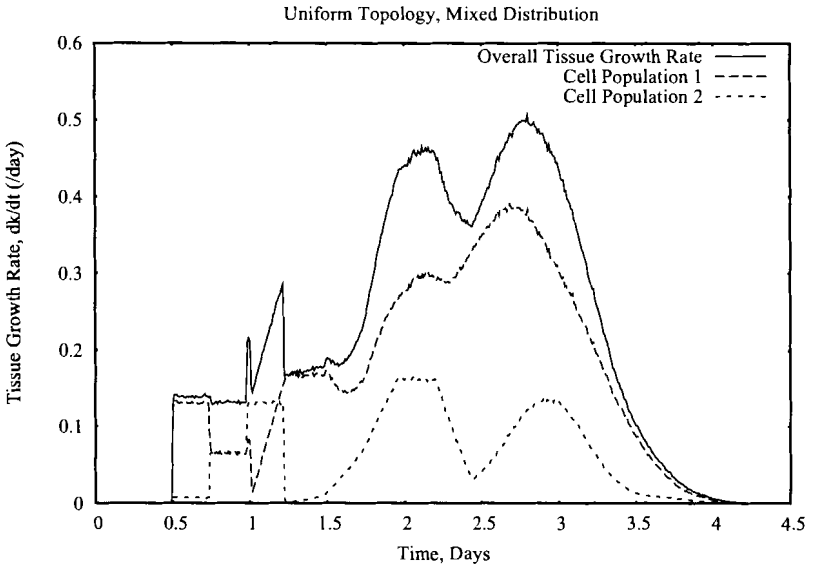


(a) Sequential Results

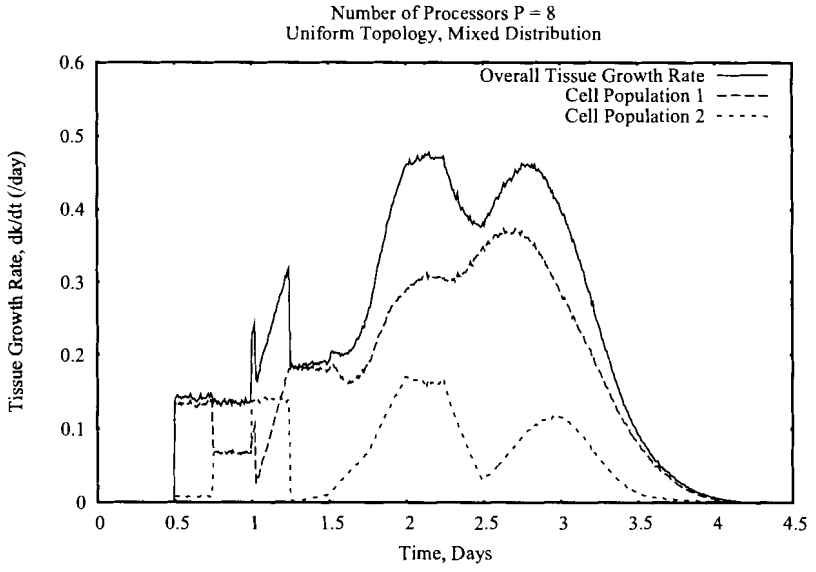


(b) Parallel Results

Figure A.2: The effects of varying the total cell seeding density on the overall tissue growth rate. The ratio H is held constant at 1.

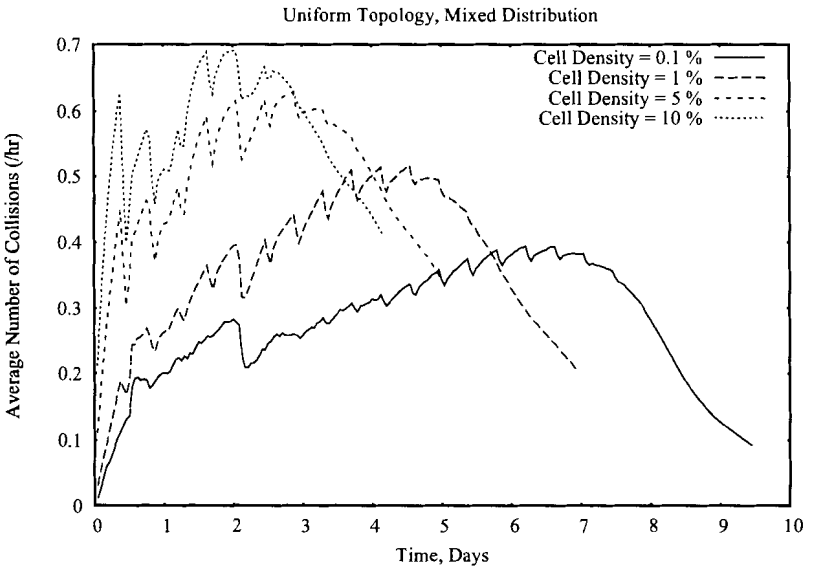


(a) Sequential Results

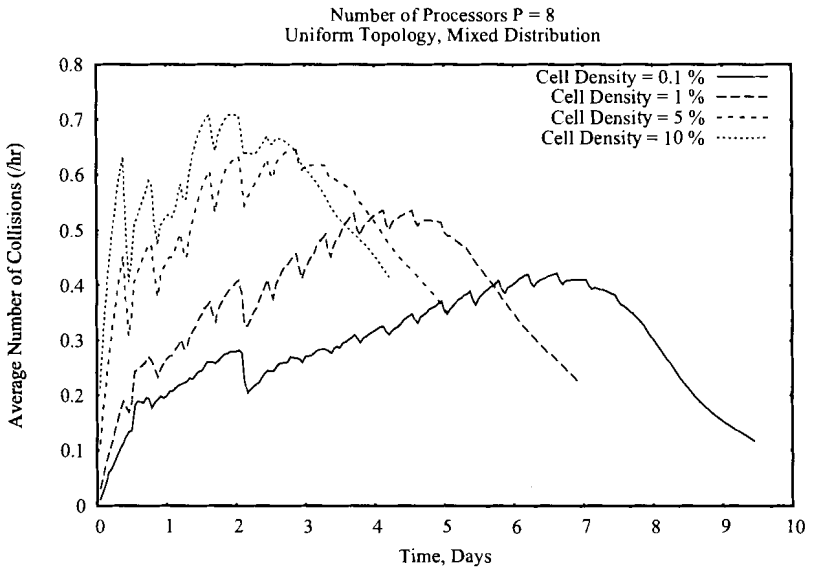


(b) Parallel Results

Figure A.3: The overall tissue growth rate is shown as the sum of the growth rate of cell population 1 and cell population 2 in the case of a seeding density of 10%.

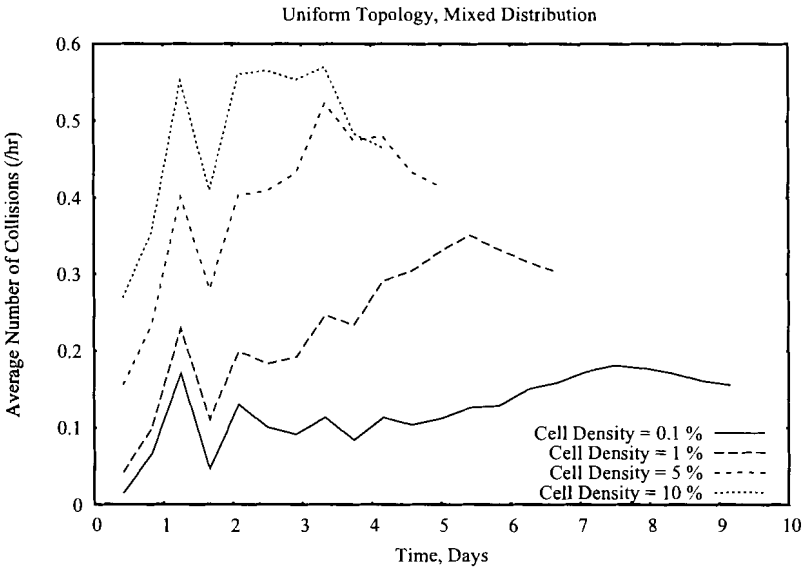


(a) Sequential Results

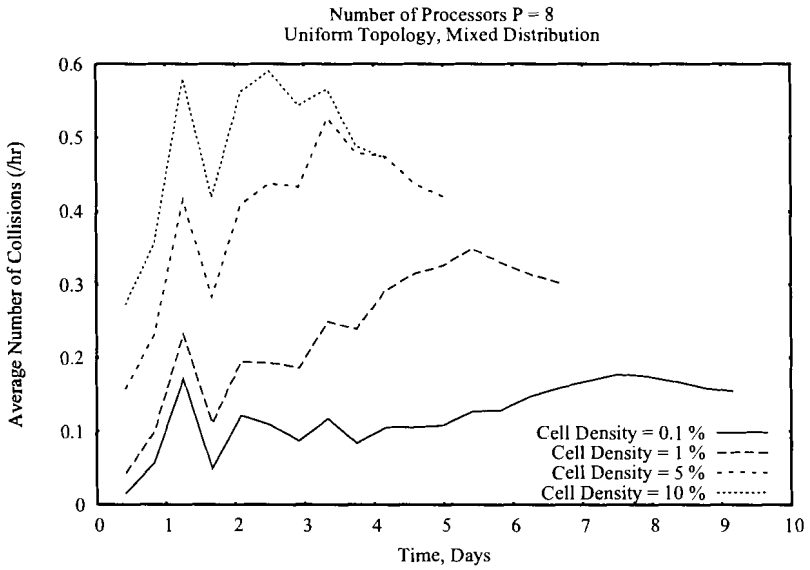


(b) Parallel Results

Figure A.4: The effects of varying cell seeding density on the average number of collisions per hour for cell population 1 for varying seeding densities.



(a) Sequential Results



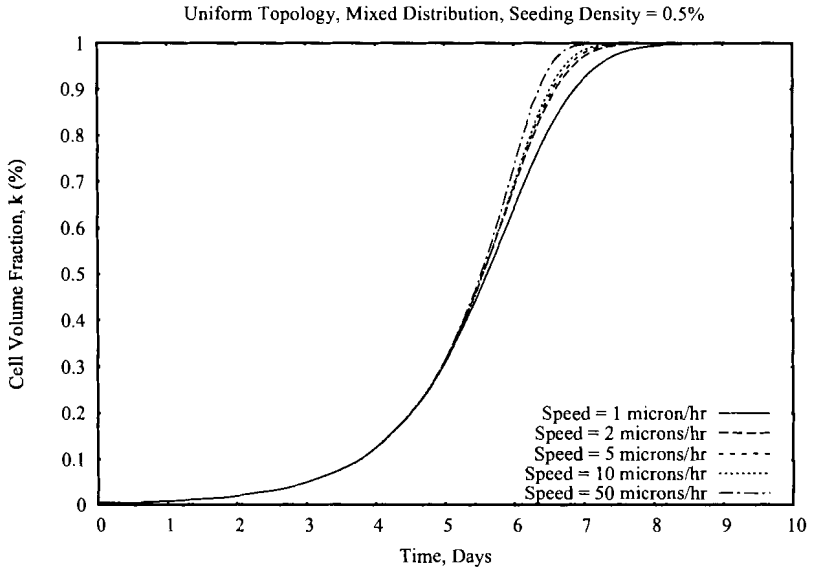
(b) Parallel Results

Figure A.5: The effects of varying cell seeding density on the average number of collisions per hour for cell population 2 for varying seeding densities.

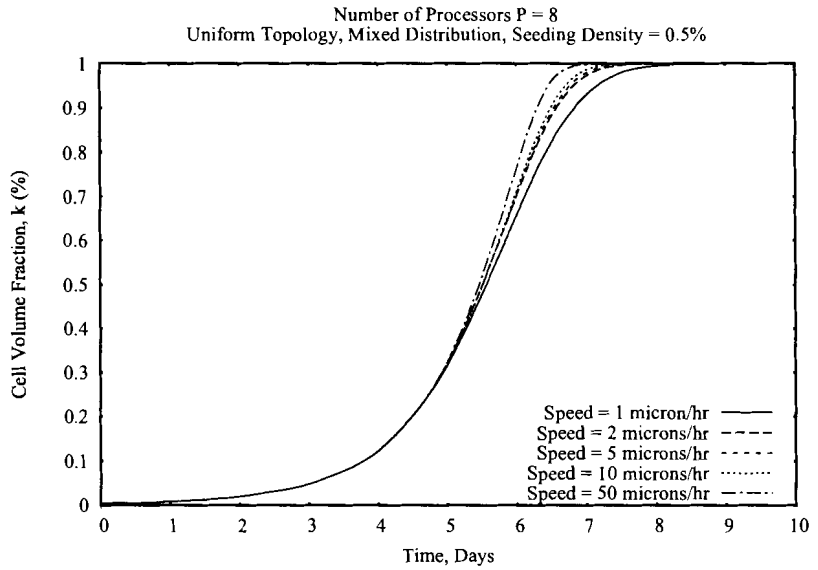
A.1.3 Effects of varying migration speed of cell population 1

The effects of varying the speed of cells on the cell volume fraction and the tissue growth rate in a mixed seeding distribution are shown in Figure A.6, Figure A.7 and Figure A.8. In these simulations, the cell speed of population 1 is assigned values of 1, 2, 5, 10, and 50 $\mu m/hr$ while the migration speed of cells in population 2 is fixed at 1 $\mu m/hr$. In addition, the total seeding density is fixed at 0.5% and the ratio H is set to 1. We observe that as the motility of cells in population 1 increases, the cell volume fraction and the overall tissue growth rate increases while confluence is attained faster. Higher motility of cells decreases the negative impact of contact inhibition on the proliferation rate because it eliminates the formation of cell colonies.

Figure A.9 and Figure A.10 show the temporal evolution of cell population 1 effective speed of locomotion. The speed of motion remains high initially when cell densities are low. An increase in cell density leads to an increase in the number of cell-cell interactions which in turn drives the effective speed of migration to lower levels.

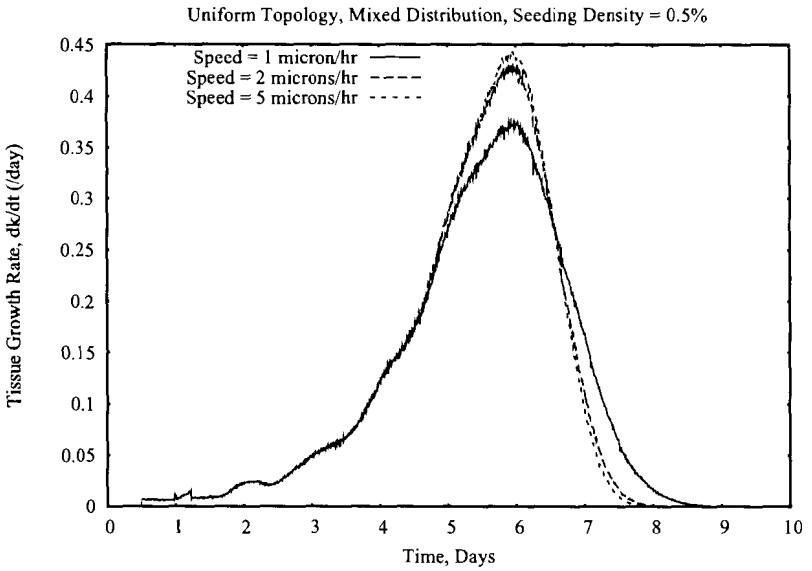


(a) Sequential Results

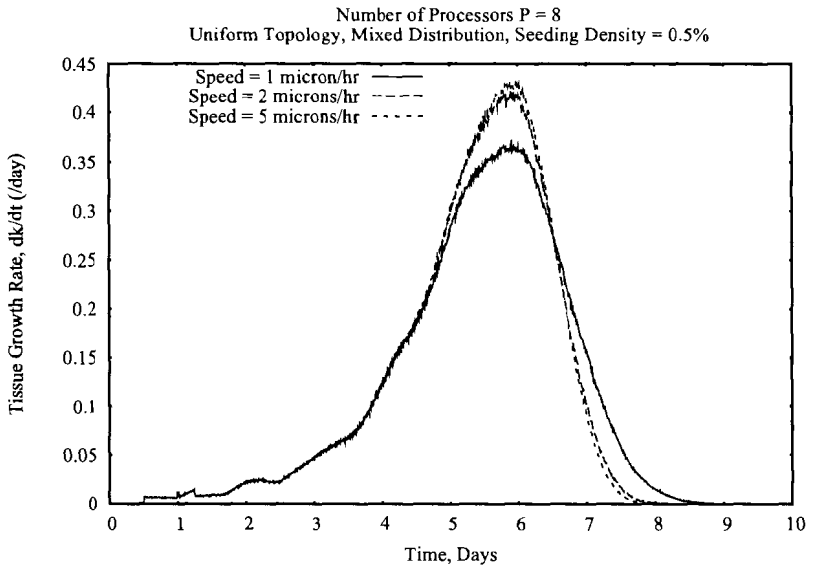


(b) Parallel Results

Figure A.6: The effects of varying the cell speed of population 1 on the cell volume fraction. Cells in population 2 move at a fixed speed of 1 $\mu m/\text{hr}$.

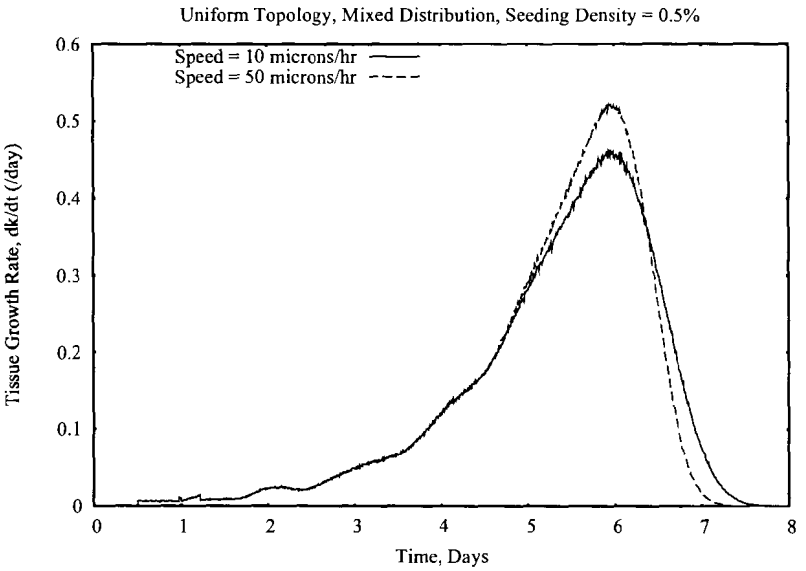


(a) Sequential Results

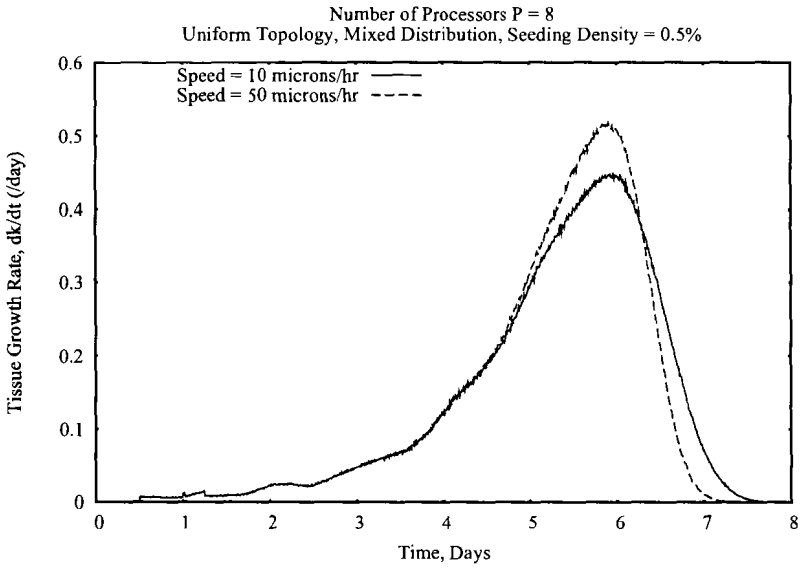


(b) Parallel Results

Figure A.7: The effects of cell motility on the overall tissue growth rate for cell speeds ranging from 1 to 5 $\mu m/hr$.

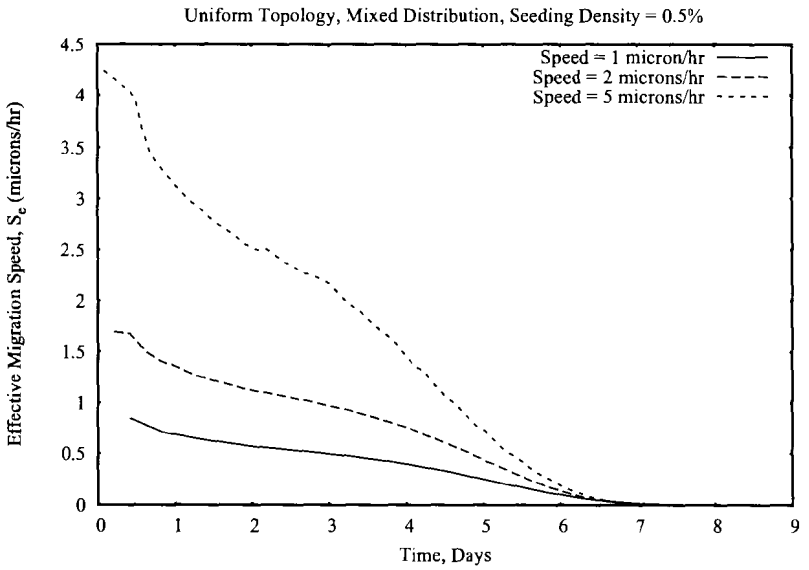


(a) Sequential Results

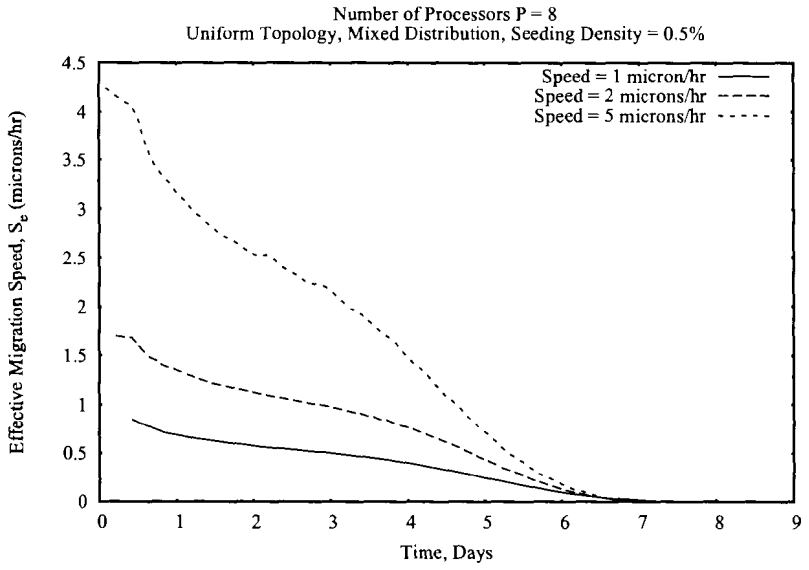


(b) Parallel Results

Figure A.8: The effects of cell motility on the overall tissue growth rate for cell speeds ranging from 10 to 50 $\mu m/hr$.

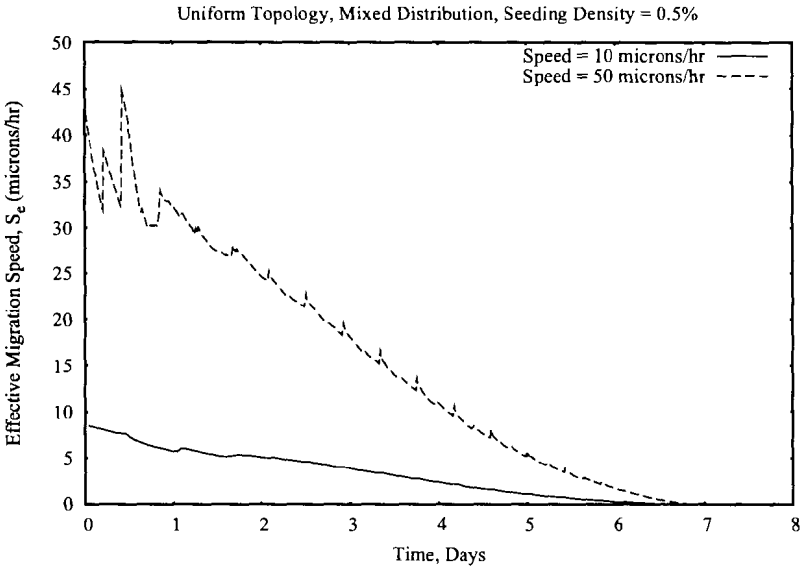


(a) Sequential Results

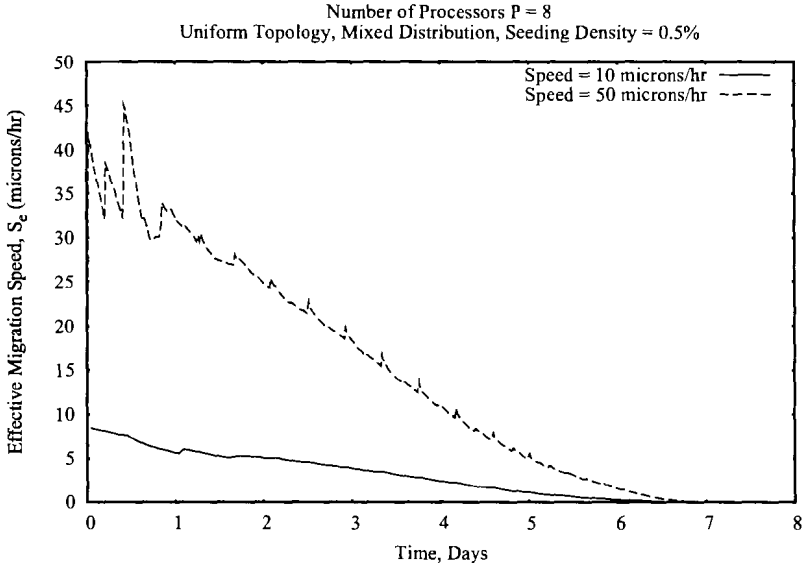


(b) Parallel Results

Figure A.9: The effects of population 1 cell motility on its effective migration speed, S_e, for cell speeds ranging from 1 to 5 μm/hr.



(a) Sequential Results

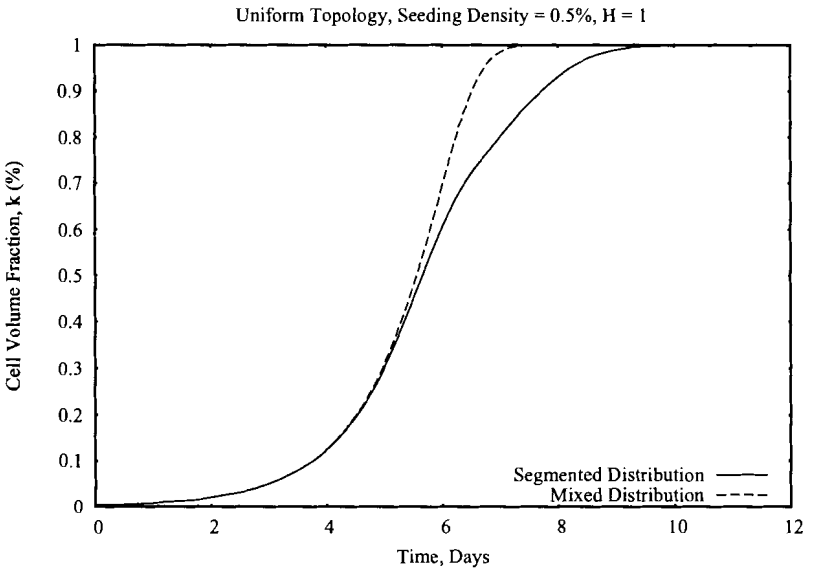


(b) Parallel Results

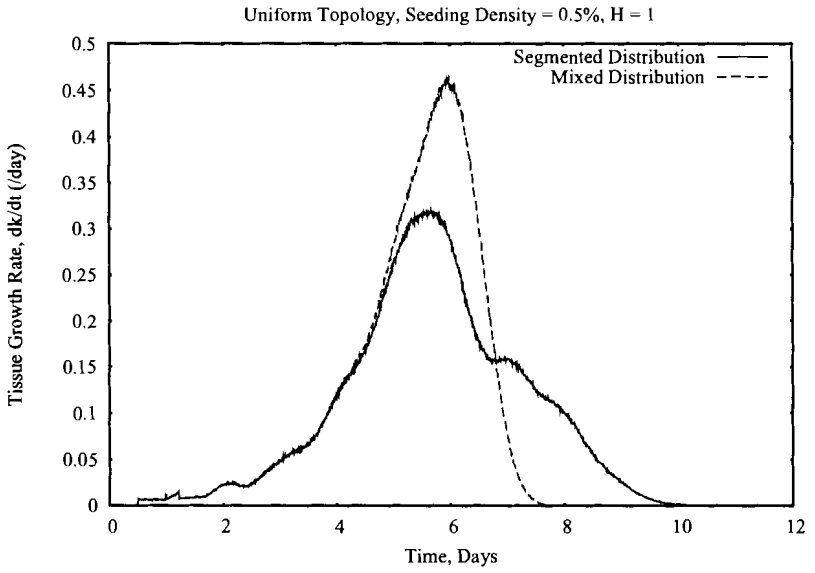
Figure A.10: The effects of population 1 cell motility on its effective migration speed, S_e, for cell speeds ranging from 10 to 50 μm/hr.

A.2 Comparison of Uniform Cell Seeding Distributions

Figures A.11 (a), A.11 (b), A.12 (a) and A.12 (b) show comparisons between the cell volume fractions and the tissue growth rates obtained by using the segmented and mixed uniform distributions. Here, $H = 1$ and a seeding density of 0.5% are used. In Figure A.11, the cells from population 1 and population 2 have a migration speed of 10 μm/hr and 1 μm/hr, respectively. While in Figure A.12 (a), the difference in the migration speed of the two cell populations is increased such that the cells from population 1 and population 2 have a migration speed of 50 μm/hr and 0.5 μm/hr, respectively. We observe that the mixed distribution takes less time to reach confluence and yields higher tissue growth rates. However, as cell speeds of population 1 increase, the difference between both seeding modes diminishes. Higher cell motility allows cells in the segmented distribution to move far away from their original seeded sites and disperse in the cellular space to minimize the impact of contact inhibition, thus creating a transient distribution that resembles more and more the one induced by the mixed distribution. Moreover, with mixed seeding, the effects of increasing cell speed are less pronounced. For instance, the tissue growth rate curves are nearly identical for both population 1 speeds when a mixed distribution is used as in both cases confluence is reached in about 8 days and their growth rate curves are very similar. Increasing population 1 speeds to very large values is more beneficial in the case of a segmented distribution where an enhanced tissue growth rate and a reduction in the time to reach confluence are both observed.

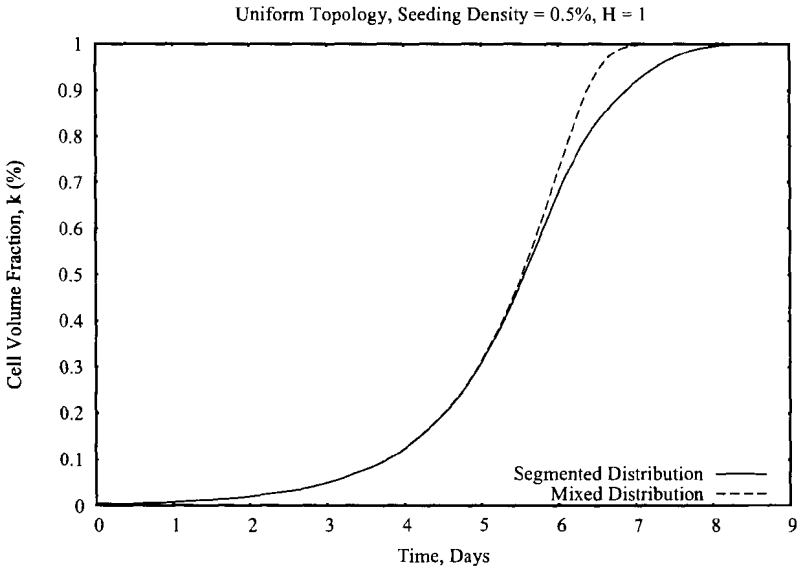


(a) Sequential Results

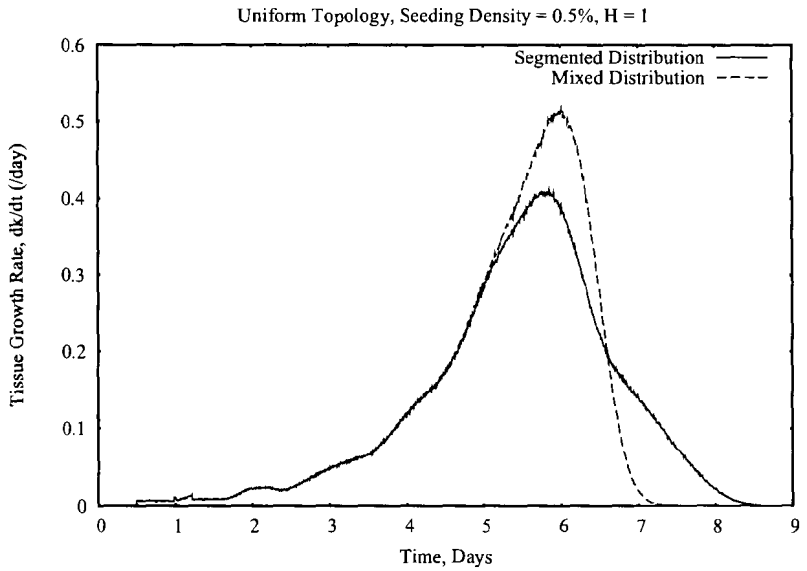


(b) Parallel Results

Figure A.11: Comparison of (a) the cell volume fraction and (b) the overall tissue growth rate for segmented and mixed seeding distributions. The cell migration speed of population 1 and population 2 is 10 μm/hr and 1 μm/hr, respectively.



(a) Sequential Results



(b) Parallel Results

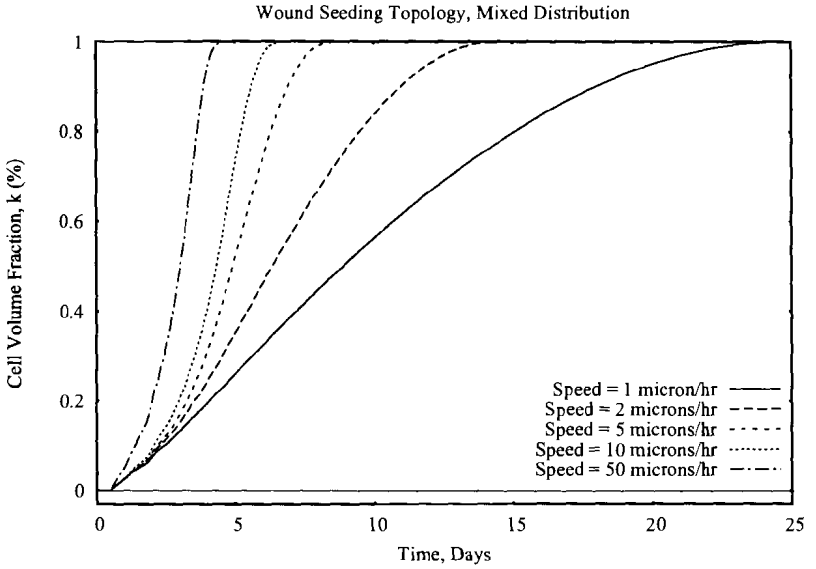
Figure A.12: Comparison of (a) the cell volume fraction and (b) the overall tissue growth rate for segmented and mixed seeding distributions. The cell migration speed of population 1 and population 2 is 50 μm/hr and 0.5 μm/hr, respectively.

A.3 Wound Seeding Topology

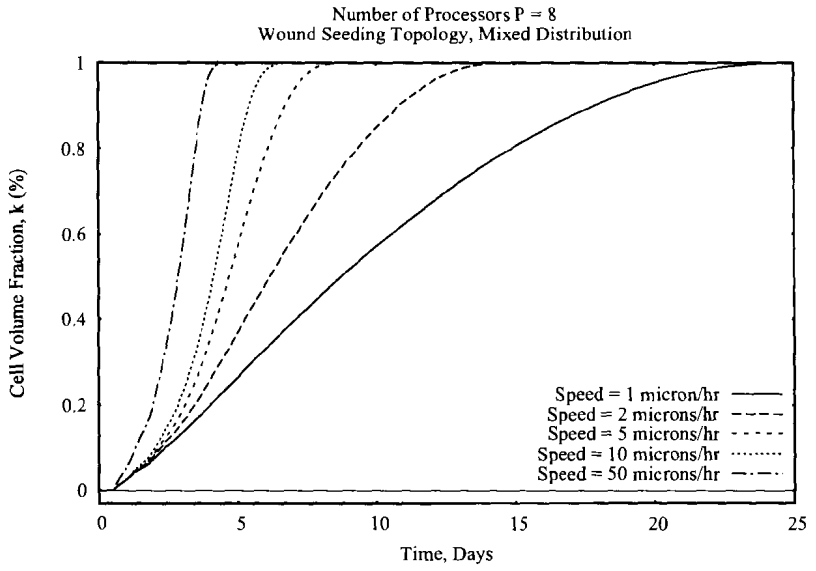
A.3.1 Mixed Distribution

A.3.2 Effects of varying migration speed of cell population 1

The proliferation rate curves in Figure A.13 demonstrate the effects of motility on the cell volume fraction in the case of a wound-seeding topology using a mixed distribution with $H = 1$. Again, in these simulation runs, the cell speed of population 1 is varied from 1, 2, 5, 10 and 50 μm/hr while the migration speed of population 2 is fixed at 1 μm/hr. We observe that as the motility of the cell increases, the proliferation rate increases and hence confluence is attained faster; thus, healing the wound. Higher motility of cells decreases the impact of contact inhibition on the proliferation rate as it reduces the formation of cell colonies. Furthermore, Figure A.14 illustrates the impact of varying the cell speed of population 1 on the overall tissue growth rate. The figure clearly depicts that increasing the cell migration speed leads to higher rates of tissue growth. Increasing cell migration speeds even to very large values continues to impact positively both the tissue growth rate and the time to reach confluence in the case of a wound seeding topology. Figure A.15 and Figure A.16 depict the temporal evolution of the effective migration speed, S_e, in the denuded area (that is, the initially empty cylinder) for different cell-population 1 speeds. At the beginning of the simulations, cells move into the "wound" at their peak speeds. The overall cell speeds drop rapidly as the "wound" becomes congested with new daughter cells and collisions become more frequent. The average speed decreases with time and shows a drastic decrease as confluence is attained due to the formation of local clusters.

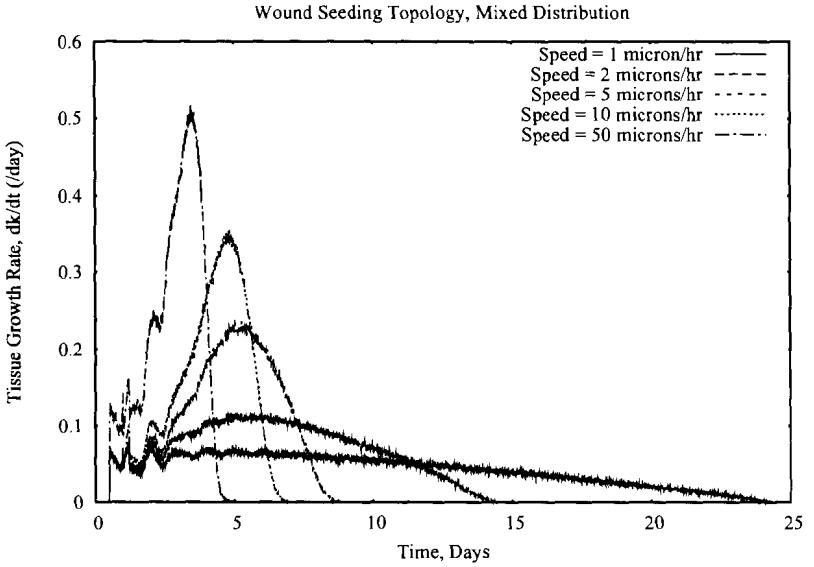


(a) Sequential Results

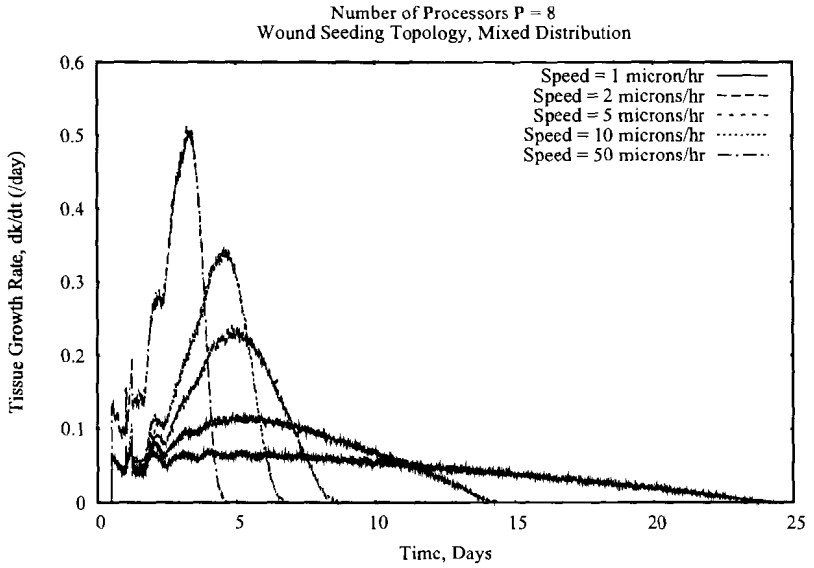


(b) Parallel Results

Figure A.13: The effects of varying the cell speed of population 1 on the cell volume fraction. Cells in population 2 move at a fixed speed of 1 $\mu m/\text{hr}$.

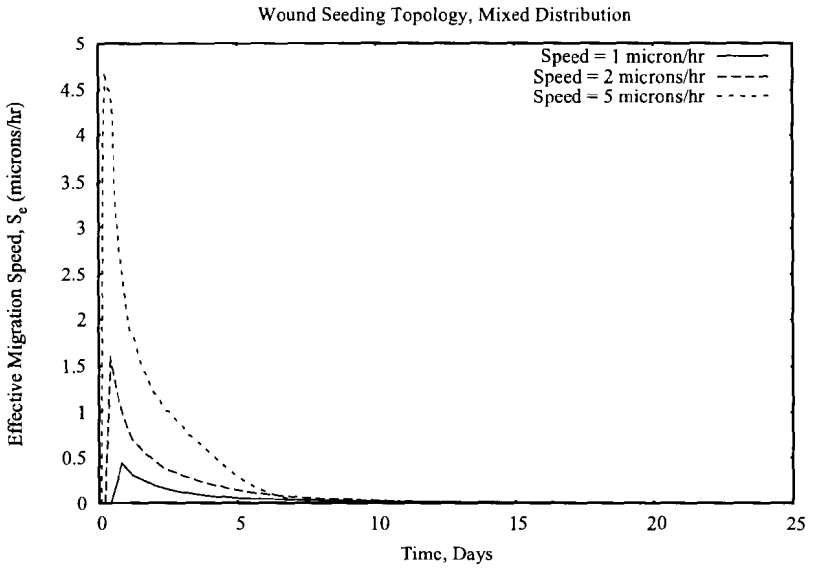


(a) Sequential Results

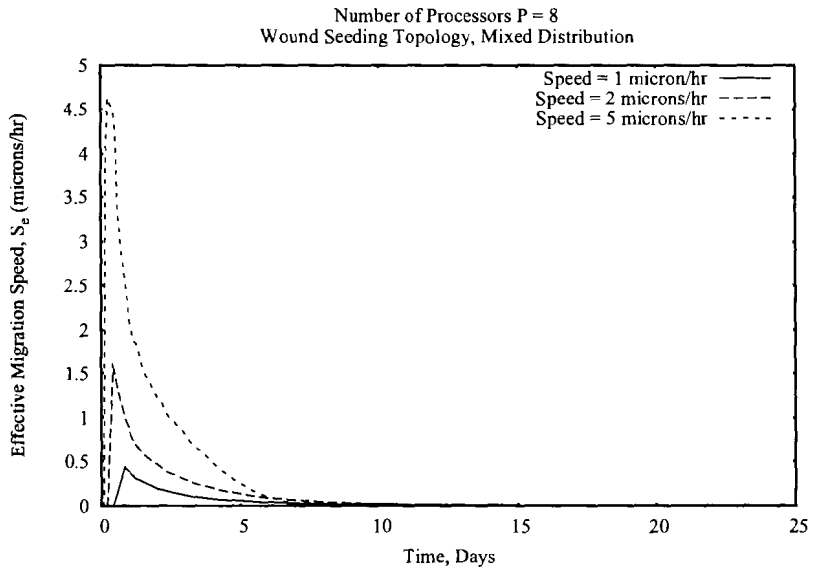


(b) Parallel Results

Figure A.14: The effects of varying the cell speed of population 1 on overall tissue growth rate. Cells in population 2 move at a fixed speed of 1 μm/hr .

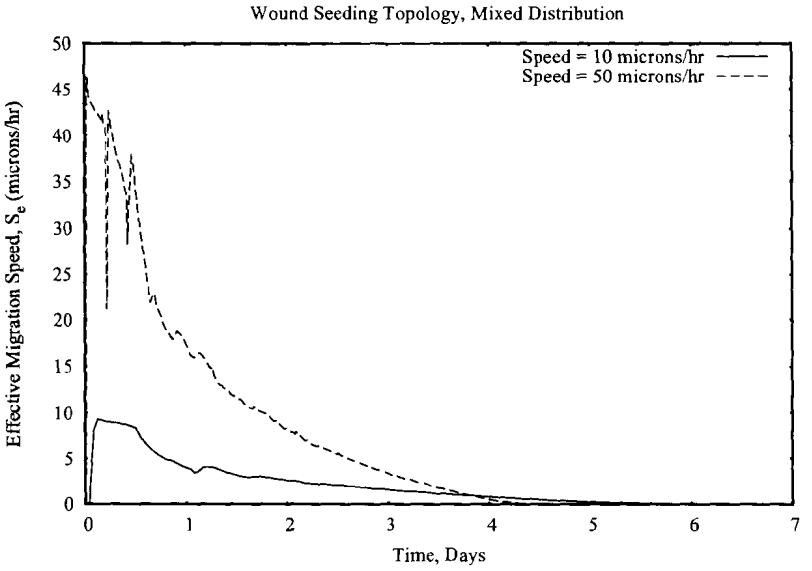


(a) Sequential Results

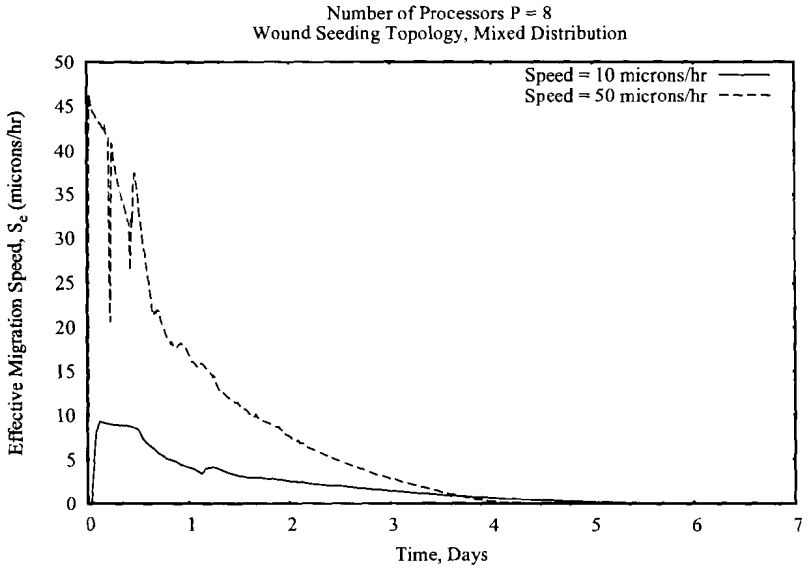


(b) Parallel Results

Figure A.15: The effects of cell motility on the effective migration speed, S_e, for (population 1) cell speeds ranging from 1 to 5 μm/hr.



(a) Sequential Results



(b) Parallel Results

Figure A.16: The effects of cell motility on the effective migration speed, S_e, for (population 1) cell speeds ranging from 10 to 50 μm/hr.

Appendix B

Glossary

In this appendix, we provide the definitions of some of the terms used throughout this thesis.

Anchorage-dependant. This term describes cells that proliferate when attached to inert surfaces.

Biofilm. This term refers to an aggregation of cells grouped by a protective adhesive matrix.

Cell migration. This term denotes the active movement of a cell from a position to another. This term is used interchangeably with *cell locomotion*, and *cell motility*.

Cell locomotion. see *cell migration*

Cell motility. see *cell migration*

Cell proliferation. This term describes the growth of cells in a population through cell division.

Confluence. This term denotes that cells occupy all (or nearly all as in the case of 99.99% confluence) of the available growth surface.

Contact inhibition. This term describes the inhibition of cell motility and growth following contact with other cells.

Endothelial cell. This term denotes a type of a mammalian cell which lines the entire circulatory system and also forms a thin lining around blood vessels.

Mitotic cycle. This term describes the process in which a single mother cell duplicates its chromosomes, and separates into two identical daughter cells.

Bibliography

[1] B. Alberts, D. Bray, J. Lewis, M. Raff, K. Roberts, and J. D. Watson. *Molecular biology of the cell.* Garland Science Publishing, New York, fourth edition, 2002.

[2] M. R. Alison and C. E. Sarraf. *Understanding cancer: From basic science to clinical practice.* Cambridge University Press, New York, 1997.

[3] B. Ben Youssef. *Cell proliferation and migration: 3-D modelling using cellular automata, development of a parallel algorithm and its parallel implementation on an IBM SP2.* Ph.D. dissertation, University of Houston, May 1999.

[4] B. Ben Youssef. A three-dimensional stochastic model for tissue growth. In *Proceedings of the 16th IASTED International Conference on Modelling and Simulation (MS 2005)*, pages 136–142, May 2005.

[5] B. Ben Youssef, G. Cheng, K. Zygourakis, and P. Markenscoff. Parallel implementation of a cellular automaton modeling the growth of three-dimensional tissues. *International Journal of High Performance Computing Applications*, 21(2):196–209, Summer 2007.

[6] B. Ben Youssef, P. Markenscoff, and K. Zygourakis. Parallelization of a 3-D computational model for wound healing. *WSEAS Transactions on Computers*, 3(4):993–998, October 2004.

[7] P. Bratley, B. L. Fox, and L. E. Schrage. *A guide to simulation.* Springer-Verlag, second edition, 1987.

[8] B. Cabral, N. Cam, and J. Foran. Accelerated volume rendering and tomographic reconstruction using texture mapping hardware. In *ACM Symposium on Volume Visualization '94*, pages 91–98, 1994.

[9] I. Chang, E. S. Gilbert, N. Eliashberg, and J. D. Keasling. A three-dimensional, stochastic simulation of biofilm growth and transport-related factors that affect structure. *Microbiology*, 149(10):2859–2871, 2003.

[10] G. Cheng, B. Ben Youssef, P. Markenscoff, and K. Zygourakis. Cell population dynamics modulate the rates of tissue growth processes. *Biophysical Journal*, 90(3):713–724, February 2006.

[11] R. S. Cherry and E. T. Papoutsakis. Modelling of contact-inhibited animal cell growth on flat surfaces and spheres. *Biotechnology and Bioengineering*, 33:300–305, January 1989.

[12] T. M. Cickovski, C. Huang, R. Chaturvedi, T. Glimm, H. G. E. Hentschel, M. S. Alber, J. A. Glazier, S. A. Newman, and J. A. Izaguirre. A framework for three-dimensional simulation of morphogenesis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2(4):273–288, 2005.

[13] J. Demongeot, J. Bezy-Wendling, J. Mattes, P. Haigron, N. Glade, and J. L. Coatrieux. Multiscale modeling and imaging: The challenges of biocomplexity. *Proceedings of the IEEE*, 91(10):1723–1737, October 2003.

[14] M. Dvir-Ginzberg, I. Gamlieli-Bonshtein, R. Agbaria, and S. Cohen. Liver tissue engineering within alginate scaffolds: effects of cell-seeding density on hepatocyte viability, morphology, and function. *Tissue Engineering*, 9(4):757 –766, August 2003.

[15] R. Dyson. *Essentials of cell biology.* Allyn and Bacon, Inc., 1978.

[16] S. P. Forestell, B. J. Milne, and L. A. Behie. A cellular automaton model for the growth of anchorage-dependent mammalian cells used in vaccine production. *Chemical Engineering Science*, 47(9-11):2381–2386, 1992.

[17] G. C. Fox, M. A. Johnson, G. A. Lyzenga, S. W. Otto, J. K. Salmon, and D. W. Walker. *Solving problems on concurrent processors. Vol. 1: General techniques and regular problems.* Prentice-Hall, Inc., Upper Saddle River, NJ, 1988.

[18] K. K. Frame and W. S. Hu. A model for density-dependent growth of anchorage-dependent mammalian cells. *Biotechnology and Bioengineering*, 32:1061–1066, October 1988.

[19] M. H. Gail and C. W. Boone. The locomotion of mouse fibroblasts in tissue culture. *Biophysical Journal*, 10(10):980–993, 1970.

[20] A. Grama, A. Gupta, G. Karypis, and V. Kumar. *Introduction to parallel computing.* Addison-Wesley, second edition, 2003.

[21] R. Gregory, R. Paton, J. Saunders, and Q. H. Wu. Parallelising a model of bacterial interaction and evolution. *BioSystems*, 76(1-3):121–131, August-October 2004.

[22] L. G. Griffith and G. Naughton. Tissue engineering: Current challenges and expanding opportunities. *Science*, 295:1009–1014, 2002.

[23] H. Gruler and B. D. Bultmann. Analysis of cell movement. *Blood Cells*, 10(1):61–77, 1988.

[24] K. A. Hawboldt, N. Kalogerakis, and L. A. Behie. A cellular automaton model for microcarrier cultures. *Biotechnology and Bioengineering*, 43(1):90–100, January 1994.

[25] C. Hecker, D. Roytenberg, J.-R. Sack, and Z. Wang. System development for parallel cellular automata and its applications. *Future Generation Computer Systems*, 16(2-3):235–247, 1999.

[26] L. L. Hench and J. M. Polak. Third-generation biomedical materials. *Science*, 295(5557):1014–1017, 2002.

[27] G. Jerusalem, Y. Beguin, M. F. Fassotte, T. Belhocine, R. Hustinx, P. Rigo, and G. Fillet. Early detection of relapse by whole-body positron emission tomography in the follow-up of patients with hodgkin's disease. *Annals of Oncology*, 14(1):123–130, January 2003.

[28] A. R. Kansal, S. Torquato, G. R. Harsh IV, E. A. Chiocca, and T. S. Deisboeck. Simulated brain tumor growth dynamics using a three-dimensional cellular automaton. *Journal of Theoretical Biology*, 203(4):367–382, 2000.

[29] S. Karlin and H. M. Taylor. *A first course in stochastic processes.* Academic Press, Inc., second edition, 1975.

[30] N. T. Karonis. Timing parallel programs that use message passing. *Journal of Parallel and Distributed Computing*, 14(1):29–36, 1992.

[31] G. Karp. *Cell biology.* McGraw-Hill, New York, second edition, 1984.

[32] S. Kouvroukoglou, C. Lakkis, D. Wallace, K. Zygourakis, and D. Epner. Bioenergetics of rat prostate cancer cell migration. *The Prostate*, 34(2):137–144, 1998.

[33] R. Langer and J. P. Vacanti. Tissue engineering. *Science*, 260:920–926, 1993.

[34] A. M. Law. *Simulation modelling & analysis.* McGraw-Hill, Inc., fourth edition, 2007.

[35] Y. Lee, S. Kouvroukoglou, L. V. McIntire, and K. Zygourakis. A cellular automaton model for the proliferation of migrating contact-inhibited cells. *Biophysical Journal*, 69(10):1284–1298, October 1995.

[36] Y. Lee, P. A. Markenscoff, L. V. McIntire, and K. Zygourakis. Characterization of endothelial cell locomotion using a markov chain model. *Biochemistry and Cell Biology*, 73:461–472, 1995.

[37] Y. Lee, L. V. McIntire, and K. Zygourakis. Analysis of endothelial cell locomotion: Differential effects of motility and contact inhibition. *Biotechnology and Bioengineering*, 43(7):622–634, March 1994.

[38] D. H. Lehmer. Mathematical methods in large-scale computing units. In *a Second Symposium on Large-Scale Digital Calculating Machinery*, volume 26 of *The Annals of the Computational Laboratory of Harvard University*, pages 141–146. Harvard University Press, 1951.

[39] C. Leopold. *Parallel and distributed computing: A survey of models, paradigms, and approaches.* John Wiley & Sons, Inc., New York, 2001.

[40] E. Lepekhin, B. Grøn, V. Berezin, E. Bock, and E. Dabelsteen. Differences in motility pattern between human buccal fibroblasts and periodontal and skin fibroblasts. *European Journal of Oral Sciences*, 110(1):13–20, January 2002.

[41] J. H. F. Lim and G. A. Davies. A stochastic model to simulate the growth of anchorage dependent cells on flat surfaces. *Biotechnology and Bioengineering*, 36:547–562, September 1990.

[42] X. Liu and P. X. Ma. Polymeric scaffolds for bone tissue engineering. *Annals of Biomedical Engineering*, 32(3):477–486, March 2004.

[43] A. F. Marée and P. Hogeweg. How amoeboids self-organize into a fruiting body: multicellular coordination in dictyostelium discoideum. *Proceedings of the National Academy of Sciences*, 98(7):3879–3883, March 2001.

[44] P. Martin. Wound healing - aiming for perfect skin regeneration. *Science*, 276:75–81, 1997.

[45] R. L. Mauck, C. C. Wang, E. S. Oswald, G. A. Ateshian, and C. T. Hung. The role of cell seeding density and nutrient supply for articular cartilage tissue engineering with deformational loading. *Osteoarthritis and Cartilage*, 11(12):879–890, December 2003.

[46] D. J. Mooney and A. G. Mikos. Growing new organs. *Scientific American*, 280:60–65, 1999.

[47] P. B. Noble and M. D. Levine. *Computer-assisted analyses of cell locomotion and chemotaxis.* CRC Press, Inc., Boca Raton, FL., 1986.

[48] B. O. Palsson and S. N. Bhatia. *Tissue engineering.* Pearson Prentice Hall, Upper Saddle River, NJ., 2004.

[49] C. M. Pancake. Is parallelism for you? *IEEE Computational Science & Engineering*, 3(2):18–37, Summer 1996.

[50] K. S. Park and K. W. Miller. Random number generators: Good ones are hard to find. *Communications of the ACM*, 31(10):1192–1201, October 1988.

[51] M. J. Quinn. *Parallel programming in C with MPI and OpenMP.* McGraw-Hill, New York, 2004.

[52] A. Ratcliffe and L. E. Niklason. Bioreactors and bioprocessing for tissue engineering. *Annals of the New York Academy of Sciences*, 961(1):210–215, 2002.

[53] M. V. Risbud, E. Karamuk, R. Moser, and J. Mayer. Hydrogel-coated textile scaffolds as three-dimensional growth support for human umbilical vein endothelial cells (HUVECs): possibilities as coculture system in liver tissue engineering. *Cell Transplantation*, 11(4):369–377, November 2002.

[54] R. C. Ruaan, G. J. Tsai, and G. T. Tsao. Monitoring and modelling density dependent growth of anchorage-dependent cells. *Biotechnology and Bioengineering*, 41:380–389, February 1993.

[55] S. Saini and T. M. Wick. Concentric cylinder bioreactor for production of tissue engineered cartilage: effect of seeding density and hydrodynamic loading on construct development. *Biotechnology Progress*, 19(2):510–521, April 2003.

[56] H. Shin, S. Jo, and A. G. Mikos. Biomimetic materials for tissue engineering. *Biomaterials*, 24(24):4353–4364, November 2003.

[57] D. Soll and D. Wessels. *Motion analysis of living cells: Techniques in modern biomedical microscopy.* Wiley-Liss, New York, 1998.

[58] D. Talia. Cellular processing tools for high-performance simulation. *Computer*, 33(9):44–52, 2000.

[59] L. Tang and B. Ben Youssef. A 3-D computational model for multicellular tissue growth. In *Proceedings of International Symposium on Biomedical Simulation (ISBMS 2006), Lecture Notes in Computer Science (LNCS)*, volume 4072, pages 29–39, July 10 -11, 2006.

[60] S. Walenta and W. F. Mueller-Klieser. Lactate: Mirror and motor of tumor malignancy. *Seminars in Radiation Oncology*, 14(3):267–274, July 2004.

[61] S. Wolfe. *Introduction to cell biology.* Wadsworth Publishing Co., 1983.

[62] S. Wolfram. *Cellular automata and complexity: collected papers.* Addison-Wesley Publishing Co., 1994.

[63] L. Yang and M. Guo, editors. *High-performance computing.* John Wiley & Sons, Inc., 2006.

[64] K. Zygourakis, R. Bizios, and P. Markenscoff. Proliferation of anchorage-dependent contact-inhibited cells: I. development of theoretical models based on cellular automata. *Biotechnology and Bioengineering*, 38(5):459–470, August 1991.

[65] K. Zygourakis, P. Markenscoff, and R. Bizios. Proliferation of anchorage-dependent contact-inhibited cells. II: experimental results and validation of the theoretical models. *Biotechnology and Bioengineering*, 38(5):471–479, August 1991.