**THE UNIVERSITY OF DANANG**

**UNIVERSITY OF SCIENCE AND TECHNOLOGY**

**FACULTY OF ADVANCED SCIENCE AND TECHNOLOGY**

**GRADUATION THESIS**

**MODELING AND SIMULATION OF THE SPREAD OF SARS-COV-2 IN LUNG TISSUE WITH CELL-DEVS**

**LE QUY THANH 17PFIEV3, 122170053**

Supervisor: **Prof. CLAUDIA FRYDMAN**

**Dr. ALI AYADI**

Co-supervisors: **Assoc. Prof. NGUYEN THANH BINH**

**Da Nang, 7/2022**

**SUPERVISOR’S COMMENT**

........................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................…..........................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................…........................................................................................................................................................................................................................

**REVIEWER’S COMMENT**

........................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................…..........................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................…........................................................................................................................................................................................................................

ABSTRACT

The SARS- CoV-2 pandemic and its unprecedented global societal and economic disruptive impact has marked the third zoonotic introduction of a highly pathogenic coronavirus into the human population. Although the previous coronavirus SARS- CoV and MERS-CoV epidemics raised awareness of the need for clinically available therapeutic or preventive interventions, no treatments with proven efficacy are available. Therefore, the simulations of tissue-specific effects of primary acute viral infections like COVID-19 are essential for understanding differences in disease outcomes and optimizing therapeutic interventions. This project presents the proposed approach through the modeling and simulation of an epithelial tissue infected by viruses, a simplified cellular immune response, a spread of the virus, and cytokine concentration.

In this approach, biological knowledge about the dynamic behavior of the SARS-CoV-2 molecular mechanisms, changes in the state of epithelial cells, and the body's immune response in time are modeled by CELL-DEVS models.

ACKNOWLEDGMENT

First and foremost, I am thankful to the teachers in the Department of PFIEV, Da Nang University of Technology, who have industriously taught and imparted valuable knowledge to us in the past school years.

I would like to express my special thanks of gratitude to Assoc. Prof. Nguyen Thanh Binh enthusiastically guided and helped us throughout the project. He facilitated my study and the completion of this thesis.

On this occasion, I would like to express my deepest gratitude and special thanks to Prof. Claudia Frydman, Dr. Ali Ayadi – my mentors who despite being extraordinarily busy with their duties, took time out to hear, guide, and keep us on the correct path and allow us to carry out my project at their esteemed organization and extending during the training.

I express my deepest thanks to buddies and all members of Polytech Marseille for their careful and precious guidance which was extremely valuable for my study both theoretical and practical.

|  |  |
| --- | --- |
| DA NANG UNIVERSITY  **UNIVERSITY OF SICIENCE AND TECHNOLOGY**  FALCUTY OF ADVANCED  SCIENCE AND TECHNOLOGY | **THE SOCIALIST REPUBLIC OF VIETNAM**  Independence - Freedom - Happiness |

1. GRADUATION PROJECT REQUIREMENTS

Student Name: Le Quy Thanh Student ID: 122170053

Class: 17PFIEV3 Major: Information Technology

1. *Topic title:*

Modeling and Simulation of the spread of SARS-COV-2 in lung tissue with CELL-DEVS

1. *Project topic :* ☐*has signed intellectual property agreement for final result*
2. *Initial figure and data:*

Data is generated by students and information searched on the internet

*Content of the explanations and calculations:*

Chapter 1: INTRODUCTION

Chapter 2: SURVEY ON MODELING AND SIMULATION IN VIROLOGY

Chapter 3: THE PROPOSED METHOD AND ITS IMPLEMENTATION

Chapter 4: RESULTS AND DISCUSSION

GENERAL CONCLUSION

REFERENCES

APPENDIX

1. *Instructor:*

Prof. Claudia Frydman

Faculty of Information Technology, University of Aix-Marseille, France

Dr. Ali Ayadi

Faculty of Information Technology, University of Strassbourg, France

Assoc. Prof. Nguyen Thanh Binh

Vietnam-Korea University of Information and Communication Technology, Vietnam

1. *Date of assignment:* 7/2/2022
2. *Date of completion:* 18/6/2022

|  |  |
| --- | --- |
|  | *Đà Nẵng, date month year 2022* |
| **Head of Division**…………………. | **Instructor** |

GUARANTEE

I guarantee:

1. The contents of this senior project are performed by us following the guidance of Prof. Claudia Frydman, Dr. Ali Ayadi, and Assoc. Prof. Nguyen Thanh Binh.

2. All references, used in this senior project thesis, are quoted with the author’s name, the project’s name, time, and location to publish clearly and faithfully.

3. All invalid copies, educated statute violations or cheating will be born the full responsibility by myselves.

Students

Le Quy Thanh

LIST OF CONTENTS

[ABSTRACT i](#_Toc107892245)

[ACKNOWLEDGMENT ii](#_Toc107892246)

[GRADUATION PROJECT REQUIREMENTS iii](#_Toc107892247)

[GUARANTEE iv](#_Toc107892248)

[LIST OF CONTENTS v](#_Toc107892249)

[LIST OF FIGURES vi](#_Toc107892250)

[LIST OF TABLES viii](#_Toc107892251)

[LIST OF ABBREVIATIONS x](#_Toc107892252)

[GENERAL INTRODUCTION xi](#_Toc107892253)

[CHAPTER 1 - INTRODUCTION 1](#_Toc107892254)

[1.1. Background and Motivation (the spread of Covid-19 in lung tissue) 1](#_Toc107892255)

[***1.1.1. SARS-CoV-2 life cycle*** 1](#_Toc107892256)

[***1.1.2. Epithelial cell’s state*** 2](#_Toc107892257)

[***1.1.3. Cytokine*** 2](#_Toc107892258)

[***1.1.4. Immune cell state*** 2](#_Toc107892259)

[***1.1.5. Motivation*** 3](#_Toc107892260)

[1.2. Aim and Objectives 3](#_Toc107892261)

[***1.2.1. Aim*** 3](#_Toc107892262)

[***1.2.1. Objectives*** 3](#_Toc107892263)

[1.3. Limitations 3](#_Toc107892264)

[1.4. Approach Method 3](#_Toc107892265)

[1.5. Research Methodology 4](#_Toc107892266)

[CHAPTER 2 – SURVEY ON MODELING AND SIMULATION IN VIROLOGY 7](#_Toc107892267)

[2.1. Modeling & simulation theory 7](#_Toc107892268)

[***2.1.1. What is a model*** 7](#_Toc107892269)

[***2.1.2. What is a simulation*** 7](#_Toc107892270)

[***2.1.3. The relation between Modeling & Simulation*** 7](#_Toc107892271)

[2.2. Introduction to simulation models 8](#_Toc107892272)

[***2.2.1. DEVS*** 8](#_Toc107892273)

[***2.1.1.1. Overview*** 8](#_Toc107892274)

[***2.1.1.2. DEVS models*** 8](#_Toc107892275)

[***2.2.2. CELL-DEVS*** 11](#_Toc107892276)

[***2.2.3. CD++ framework*** 15](#_Toc107892277)

[2.3. Literature review 18](#_Toc107892278)

[***2.3.1. Simulation approaches in virology*** 18](#_Toc107892279)

[***2.3.2. DEVS and cell DEVS simulation approaches in virology*** 19](#_Toc107892280)

[***2.3.3. Discussion (limits of previous approaches)*** 19](#_Toc107892281)

[CHAPTER 3 - THE PROPOSED METHOD AND ITS IMPLEMENTATION 20](#_Toc107892282)

[3.1. Overview of the proposed approach 20](#_Toc107892283)

[3.2. Cell-DEVS modeling of epithelial cell’s state 20](#_Toc107892284)

[***3.2.1. Definition*** 20](#_Toc107892285)

[***3.2.2. Rules*** 20](#_Toc107892286)

[***3.2.3. Measurement*** 23](#_Toc107892287)

[3.3. Cell-DEVS modeling of immune cell’s state 23](#_Toc107892288)

[***3.3.1. Definition*** 23](#_Toc107892289)

[***3.3.2. Rules*** 24](#_Toc107892290)

[***3.3.3. Measurement*** 25](#_Toc107892291)

[3.4. Cell-DEVS modeling of cytokine concentration 25](#_Toc107892292)

[***3.4.1. Definition*** 25](#_Toc107892293)

[***3.4.2. Rules*** 25](#_Toc107892294)

[***3.3.1. Quantification*** 27](#_Toc107892295)

[3.5. Cell-DEVS modeling of virus spread 27](#_Toc107892296)

[*3.5.1. Definition* 27](#_Toc107892297)

[***3.5.2. Rules*** 27](#_Toc107892298)

[***3.5.3. Quantification*** 28](#_Toc107892299)

[3.6. Combination of the previous cell DEVS models (epithelial cell’s state, immunity cell’s state, cytokine concentration, and virus propagation) 28](#_Toc107892300)

[CHAPTER 4 – RESULTS AND DISCUSSION 31](#_Toc107892301)

[4.1. Development environment 31](#_Toc107892302)

[*4.1.1. Tools requirement* 31](#_Toc107892303)

[*4.1.2. Development procedure* 31](#_Toc107892304)

[4.2. Application of the proposed cell-DEVS models to different scenarios 37](#_Toc107892305)

[*4.2.1. Scenario 1* 37](#_Toc107892306)

[*4.2.2. Scenario 2* 37](#_Toc107892307)

[*4.2.3. Scenario 3* 37](#_Toc107892308)

[4.3. Results and discussion 37](#_Toc107892309)

[*4.2.1. Scenario 1* 37](#_Toc107892310)

[*4.2.2. Scenario 2* 41](#_Toc107892311)

[*4.2.3. Scenario 3* 44](#_Toc107892312)

[GENERAL CONCLUSION 48](#_Toc107892313)

[REFERENCES 49](#_Toc107892314)

[APPENDIX 52](#_Toc107892315)

LIST OF FIGURES

[Figure 1.1: The coronavirus virion and life cycle. 1](#_Toc107877955)

[Figure 1.2: The epithelial cell’s state. 2](#_Toc107877956)

[Figure 1.3: Immune cell's state. 2](#_Toc107877957)

[Figure 1.4: Approach method. 4](#_Toc107877958)

[Figure 1.5: Previous research results in practice obtained by Sego et al. in 6](#_Toc107877959)

[Figure 2.1: DEVS atomic model semantics. 9](#_Toc107877960)

[Figure 2.2: A coupled model 10](#_Toc107877961)

[Figure 2.3: Definition of transition functions for cells with transport delays 14](#_Toc107877962)

[Figure 2.4: External and internal transition function for inertial delays models. 15](#_Toc107877963)

[Figure 2.5: A Cell-DEVS specification in CD++. 16](#_Toc107877964)

[Figure 3.1: Epithelial cell’s state in CELL-DEVS model. 21](#_Toc107877965)

[Figure 3.2: Immune cell’s state in CELL-DEVS model. 24](#_Toc107877966)

[Figure 3.3: State diagram and interactions of epithelial cell, immune cell, virus, cytokine. 30](#_Toc107877967)

[Figure 4.1: Project’s folders organization structure. 32](#_Toc107877968)

[Figure 4.2: Simulation by executing run.sh. 33](#_Toc107877969)

[Figure 4.3: The log file. 34](#_Toc107877970)

[Figure 4.4: Visualization of the web app’s interface. 34](#_Toc107877971)

[Figure 4.5: Epithelial cell’s state on the web app. 35](#_Toc107877972)

[Figure 4.6: Immune cell’s state simulation on the web app. 35](#_Toc107877973)

[Figure 4.7: Cytokine simulation on the web app. 36](#_Toc107877974)

[Figure 4.8: Immune Cell’s state simulation on the web app. 36](#_Toc107877975)

[Figure 4.9: Excel file statistics the number of characteristics. 37](#_Toc107877976)

[Figure 4.10: Characteristic quantification graphs. 37](#_Toc107877977)

[Figure 4.11: Simulation of the progression of infection in four models over time in scenario 1. 39](#_Toc107877978)

[Figure 4.12: Simulation of the progression of infection in four models over time in scenario 1. 40](#_Toc107877979)

[Figure 4.13: Quantification for the active immune cell during simulation time in scenario 1. 40](#_Toc107877980)

[Figure 4.14: Quantification for the cytokine during simulation time in scenario 1. 41](#_Toc107877981)

[Figure 4.15: Quantification for the virus in the extracellular environment during simulation time in scenario 1. 41](#_Toc107877982)

[Figure 4.16: Simulation of the progression of infection in four models over time in scenario 2. 42](#_Toc107877983)

[Figure 4.17: Simulation of the progression of infection in four models over time in scenario 2. 43](#_Toc107877984)

[Figure 4.18: Quantification for the active immune cell during simulation time in scenario 2. 44](#_Toc107877985)

[Figure 4.19: Quantification for the cytokine during simulation time in scenario 2. 44](#_Toc107877986)

[Figure 4.20: Quantification for the virus in the extracellular environment during simulation time in scenario 2. 45](#_Toc107877987)

[Figure 4.21: Simulation of the progression of infection in four models over time in scenario 3. 46](#_Toc107877988)

[Figure 4.22: Simulation of the progression of infection in four models over time in scenario 3. 46](#_Toc107877989)

[Figure 4.23: Quantification for the active immune cell during simulation time in scenario 3. 47](#_Toc107877990)

[Figure 4.24: Quantification for the cytokine during simulation time in scenario 3. 47](#_Toc107877991)

[Figure 4.25: Quantification for the virus in the extracellular environment during simulation time in scenario 3. 48](#_Toc107877992)

LIST OF TABLES

[Table 1.1: Parameter in rules.. 5](#_Toc42343632)

LIST OF ABBREVIATIONS

|  |  |
| --- | --- |
| **Abbreviations** | **Explanations** |
| SARS-CoV-2 | Severe acute respiratory syndrome coronavirus 2 |
| MERS-CoV | The Middle East Respiratory Syndrome Coronavirus |
| COVID-19 | Coronavirus Disease first appeared in 2019 |
| RNA | Acid Ribonucleic |
| ODE | An ordinary differential equation |
| DEVS | Discrete Event System Specification |
| CELL-DEVS | A combination of Cellular Automata and Discrete Events Systems Specifications |
| CA | Cellular Automata |
|  |  |

GENERAL INTRODUCTION

The current global pandemic of COVID-19, caused by the novel coronavirus Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2), has motivated the study of beta coronavirus diseases at multiple spatial and temporal computational modeling scales [1]. The time course, the severity of symptoms, and complications from SARS-CoV-2 infection are highly variable from patient to patient [2]. Mathematical modeling methods integrate the available host- and pathogen-level data on disease dynamics that are required to understand the complex biology of infection and immune response to optimize therapeutic interventions [3–5].

Building multiscale models of acute primary viral infection requires the integration of submodels of multiple biological components across scales (e.g., viral replication and internalization, immune system responses). In the context of viral infection dynamics, specialized ODE models can describe both the entire virus-host response at the tissue and organ levels and different stages of the viral replication cycle within cells, such as binding and internalization [6,7], viral genome replication and translation [8,9], assembly, packaging and release [10,11]. By fitting ODE models to clinical or experimental data, researchers have been able to estimate important parameters, such as the turnover rate of target cells, average lifetimes of viral particles and infected cells, and the rate of production of new viral particles by infected cells [12]. Although these theoretical studies on infectious diseases are useful, they are difﬁcult to apply in practice. Speciﬁcally, they have shortcomings for deﬁning contact processes, the behavior of the individual,s and the spatial dimension in the model. Cellular automata (CA) allows for the development models that overcome the above-mentioned shortcomings.

Although CA has been successfully applied to develop disease spread models, its discrete-time nature considers time as isomorphic to the natural numbers set N (i.e., time advances at constant steps). Therefore, all cell states that are supposed to happen between timesteps must be either neglected or delayed matching the simulation timestep. CA is not trivial to integrate with other models deﬁned in other formalisms, as well as deﬁning advanced timing conditions for each cell. The Cell-DEVS for-malism [13] solves these issues by combining CA and the Discrete EVent System Speciﬁcations (DEVS) [14] to describe n-dimensional cell spaces as discrete-event models.

Here I illustrate the application of Cell-DEVS to build a multi-level simulation modeconsideringer the host immune response, and cellular tissue damage in both time and space. In Chapter 1, I will introduce the project in detail such as the background knowledge related to the spread of Covid-19 in lung tissue, the goals to be achieved, the approach method and the research methodology. Chapter 2 provides the basic knowledge of modeling and simulation, especially the CELL-DEVS model as well as some of its approaches in virology. Chapter 3 will describe in detail how to implement the proposed CELL-DEVS model into the model of virus spread and the body's defense mechanism. Chapter 4 will describe possible scenarios when applying the model to export in practice and the results obtained from that, making evaluation comments.

# **CHAPTER 1 - INTRODUCTION**

## **1.1. Background and Motivation (the spread of Covid-19 in lung tissue)**

### ***1.1.1. SARS-CoV-2 life cycle***

The following sections describe the three main steps of the SARS-CoV-2 viral cycle:

1. The viral entry: To gain access to the host cell cytoplasm, the virus via its spicules (spike S) attaches to host cell receptors called angiotensin-converting enzymes 2 (ACE2). This binding leads to between viral and cellular membrane, releasing the viral RNA into the cytoplasm to start its replication process [15].

2. The viral replication: Once inside the host cell, the virus hijacks the host cell’s machinery, i.e. all production mechanisms such as transcription, translation and replication, turning it into a manufacturing facility producing multiple copies of SARS-CoV-2.

3. The viral export: The genomic RNA and its structural proteins (M, S, E and N) are assembled creating a viral nucleocapsid. Then, the resulting virion is transported to the surface of the infected cell through the secretion pathway and released out of the cell by exocytosis, ready to infect other cells [16].

Diagram

Description automatically generated

Figure 1.1: The coronavirus virion and life cycle [17].

### ***1.1.2. Epithelial cell’s state***

The change in cell type during the spread of the virus is shown in the figure below:

Diagram, venn diagram

Description automatically generated

Figure 1.2: The epithelial cell’s state.

Epithelial cells are of type uninfected if they have not yet internalized any virus. They are of type infected if they have internalized virus, they can be in the process of viral replication or they can not generate viruses. And after a certain time, these infected cells will die whether they can produce a virus or not.

### ***1.1.3. Cytokine***

Cytokines are a broad and loose category of small proteins important in cell signaling.

Once infected, epithelial cells secrete signaling molecules to alert the immune system and complement proteins as warning signals to neighboring cells [39,40]. Some of these cytokines, like Type 1 interferons, can induce autocrine and paracrine anti-viral responses (e.g., inhibiting viral replication, viral entry or inducing cell death) [41].

Locally, exposure to cytokine signaling results in the activation of immune cells. Upon activation, immune cells migrate towards infection sites guided by cytokine. Lastly, activated immune cells amplify the immune signal by secreting additional cytokines into the extracellular environment.

### ***1.1.4. Immune cell’s state***

The change in immune type during the spread of the virus is shown in the figure below:

A picture containing text, antenna

Description automatically generated

Figure 1.3: Immune cell's state.

*Figure STYLEREF 1 \s 1. SEQ Figure \\* ARABIC \s 1 3:* Immune cell state

As the viral load increases, immune signaling increases rapidly (this increase is associated with the onset of fever and other symptoms) recruiting more circulating cells of the innate immune system to the infection site. Immune signals from infected cells and innate immune cells help trigger the adaptive immune response.

Immune cells will be activated when there is a large number of cytokines around. Immune cells can also be inactivated after a period of activation.

### ***1.1.5. Motivation***

The Covid-19 pandemic has caused great damage to the world and currently, there is no treatment with proven effectiveness. I believe that simulating the spread of the virus in lung tissue in this project will help biologists build hypothetical scenarios from which to develop treatments and better understand virus behavior.

## **1.2. Aim and Objectives**

### ***1.2.1. Aim***

Simulate the spread of the virus in lung tissue based on the relevant knowledge learned, thereby defining characteristic parameters to customize to suit each different infection situation in reality. This simulation will perform through four models linked together by logical relationships including the epithelial cell state model, virus propagation model, immune cell model and the cytokine concentration model.

### ***1.2.1. Objectives***

* Gain knowledge from the literature about the spread of the virus, the factors that influence it, and the body's response to viral entry;
* Create a model of epithelial cell state. virus propagation, immune cell and cytokine concentration using rules extracted from the acquired knowledge;
* Combine these models by logical relationships to obtain a complete model;
* Simulate 3 real virus infection scenarios, evaluate and gain experiences to improve models.

## **1.3. Limitations**

The project had to be completed in 2 months, leading to many shortcomings in the steps. The CELL-DEVS model is easy to implement but the accuracy will not be high compared to other complex mathematical models like ODE.

## **1.4. Approach Method**

The approach method is shown in sequence in the figure below:

Diagram

Description automatically generated

Figure 1.4: Approach method.

First, from theories that have been obtained from studies of the spread of the virus in lung cells as well as the body's response. I make hypotheses that are quantified by reasoning rules to build interconnected models based on the CELL-DEVS model. Then simulate them according to real scenarios, observe the results and make changes in the hypothesis to improve the performance of the model more like reality. Therefore, I will go through a loop that includes hypothesized, simulation, observation, and hypothesis changes until the results are close to reality.

## **1.5. Research Methodology**

From the knowledge extracted from the paper, I assume the rules to create the models. These rules will contain parameters, divided into 2 types: independent (will not change the value in any scenario) and dependent (will change value when the scenario changes). Details are described in the table below:

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Value** | **Description** |
| Virion released | 1-100 | The number of virions that the epithelial cell secretes each time when it is in the secreting state |
| Immune signaling of the infected cell | 1 | A parameter represents the intensity of the immune request signal that the infected cell sends to the host body |
| Immune signaling of immune cell | 1.2 | A parameter represents the intensity of the immune request signal that the immune cell sends to the host body |
| Virus attached to the cell’s surface | 20% | The percentage of virus that is in a given epithelial cell that continues to attach to this cell after most of the virus has migrated to other cells |
| Virus moves to another cell | 80% | The percentage of viruses that are in certain epithelial cells migrate to other cells |
| Virus absorbed on the surface | 10% | The percentage of viruses that adhere to the surface of epithelial cells will enter the cell |
| Conversion from immune signal to cytokine | 1000 | The conversion rate from immune signals to the number of cytokines that the body secretes |
| Cytokine attached to the cell’s surface | 10% | The percentage of cytokines that is in a given epithelial cell that continues to attach to this cell after most of the virus has migrated to other cells |
| Cytokine moves to neighbor’s cell | 90% | The percentage of cytokines that are in certain epithelial cells migrate to other cells |
| Cytokine absorbed on the surface | 10% | The percentage of cytokines that adhere to the surface of epithelial cells will enter the cells |
| Virion release time | 20 hour | The length of time that epithelial cells are secreting to release virions |
| Anti-virus time | 8 hours | The length of time that epithelial cells are non-secreting the virus |
| Cytokine release time | 8 hours | A period of time that the body secretes cytokine |
| Immune cell activation time | 10 hours | The length of time that immune cells are in an activation state against the virus |

*Table 1.1:* Parameter in rules.

Once the rules are in place, I can simulate and observe the results, comparing them with the actual research obtained by Sego et al. in [2] in the example below.

A picture containing graphical user interface

Description automatically generated

Figure 1.5: Previous research results in practice obtained by Sego et al. in [18]

Thereby continuing the loop of changing the hypothesis and evaluating the rule until it is close to reality.

# **CHAPTER 2 – SURVEY ON MODELING AND SIMULATION IN VIROLOGY**

## **2.1. Modeling & simulation theory**

### ***2.1.1. What is a model***

A model is a product (physical or digital) that represents a system of interest. A model is similar to but simpler than the system it represents while approximating most of the same salient features of the real system as close as possible. A good model is a judicious tradeoff between realism and simplicity. A key feature of a model is manipulability. A model can be a physical model (for example a physical architectural house scale model, a model aircraft, a fashion mannequin, or a model organism in biology research); or a conceptual model (for example a computer model, a statistical or mathematical model, a business model).

### ***2.1.2. What is a simulation***

A simulation is a form of computer modeling. Although the term simulation also applies to physical models such as flight simulators or role-playing games, the most common definition of simulation refers to the process of translating a conceptual model of some system into a computer model of the system. As with modeling in general, a conceptual model can range from a simple mathematic equation scrawled on the back of an envelope to a complex natural language description spanning several volumes. A simulation model represents key features of a conceptual model through a software program. This computer model takes data as input, executes its underlying software programming, and produces output data for analysis.

A simulation model represents a system. A system is a set of real or hypothetical elements interconnected in a manner that gives the system an overall identity and behavior. Systems can range from the atoms in a molecule to people in a city to galaxies in the universe.

In creating a model of any system, the modeler must describe the germane characteristics of system elements and capture important relationships among these elements. The modeler must also describe system boundaries because a system almost always exists within a larger environment. Regional economies or ecosystems are self-contained in some senses, for example, but they have connections to larger economies and environments, respectively.

Simulation offers both advantages and challenges to modelers, discussed below, but in general, it is a useful way to extend knowledge of real or hypothetical systems. Simulation allows the modeler go beyond collecting and analyzing data about a system by using these data to build models of the system on a computer. The use of simulation is growing given the increasing amount of digital data, the ready availability of computers, improvements in programming languages and simulation methods, and the increasing numbers of supporting applications such as geographic information systems.

### ***2.1.3. The relation between Modeling & Simulation***

Simulations allow evaluating a model to optimize system performance or to make predictions about a real system. Simulations are useful to study the properties of a model of a real-life system that would otherwise be too complex, too large/small, too fast/slow, not accessible, too dangerous, or unacceptable to engage in. While a model aims to be true to the system it represents, a simulation can use a model to explore states that would not be possible in the original system.

## **2.2. Introduction to simulation models**

### ***2.2.1. DEVS***

#### **2.1.1.1. Overview**

DEVS [33] is a discrete event formalism proposed by Bernard P. Zeigler in the 70s [34]. Thanks to its universality property, DEVS is seen as a pivot formalism for heterogeneous formalism integration [35]. Several works have shown the integration of event formalisms like Petri Nets [36] in DEVS, but also the integration of a hybrid formalism like DEV&DESS [37] which specifies the interaction between discrete and continuous models.

Many works have pointed out the benefits of DEVS-based standard development [38]. Most of these benefits are linked to the formal properties of this formalism. Universality and unicity: Every event system can be described in DEVS, and this description is unique. Closure under coupling and abstract algorithm: DEVS defines a coupled model structure (a model composed of several DEVS submodels). It is shown that each coupled model is equivalent to an atomic model (allowing hierarchical design), and the chosen hierarchy does not impact the simulation results obtained with the abstract algorithm.

The separation between the model, the simulator, and the experimental frame: The definition of a DEVS model is independent of the simulator and the experimental frame. In the classic DEVS formalism, Atomic DEVS captures the system behavior, while Coupled DEVS describes the structure of the system.

In the classic DEVS formalism, Atomic DEVS captures the system behavior, while Coupled DEVS describes the structure of the system.

#### **2.1.1.2. DEVS models**

I can consider the devs model as a coupled model, it will be composed of several atomic or coupled submodels.

* **Atomic devs model**

An *atomic* model is specified as *M* = <*X, Y, S, δint, δext, λ, ta*>.

Diagram

Description automatically generated

Figure 2.1: DEVS atomic model semantics [19].

Where

* *X* = { (*p,v*)| *p* ∈ *IPort*s, *v* ∈*Xp*} is the set of *input* events, where *IPorts* represents the set of input ports and *Xp* represents the set of values for the input ports
* *Y* = {(*p,v*)| *p* ∈ *OPort*s, *v* ∈ *Yp*} is the set of *output* events, where *OPorts* represents the set of output ports and *Yp* represents the set of values for the output ports
* *S* is the set of sequential *states*;
* *δ*ext: *Q* × *X*  🡪 *S* is the *external* state transition function, with *Q* = {(*s,e*) / *s* ∈ *S, e* ∈ [0, *ta*(*s*)]}
* *λ*: *S* 🡪 *Y* is the *output* function
* *ta*: *S* 🡪 *R*0+ **∪** ∞ is the *time advance* function.

Figure 2.1 shows an informal depiction of DEVS atomic models.

At any given moment, a DEVS model is in a state *s* ∈ *S*. In the absence of external events, it remains in that state for a lifetime defined by *ta(s).* When *ta(s)* expires, the model outputs the value*λ(s)* through a port *y* ∈ *H*, and it then changes to a new state given by *δint(s)*. A transition that occurs due to the consumption of time indicated by *ta(s)* is called an internal transition. On the other hand, an external transition occurs due to the reception of an external event. In this case, the external transition function determines the new state, given by *δext (s, e, x),* where s is the current state, *e* is the time elapsed since the last transition, and *x* ∈ *X* is the external event that has been received.

The time advance function can take any real value between 0 and ∞. A state for which *ta(s)* = 0 is called a transient state (which will trigger an instantaneous internal transition). In contrast, if *ta(s)* = ∞, then s is said to be a passive state, in which the system will remain perpetually unless an external event is received (can be used as a termination condition).

* **A DEVS coupled model**

A DEVS coupled model is composed of several atomic or coupled submodels. It is formally defined by *CM* = <*X, Y, D, EIC, EOC, IC, select*>.

Diagram, schematic

Description automatically generated

Figure 2.2: A coupled model [20]

Where

* *X* = {(*p,v*)|*p* ∈ *IPorts*, *v* ∈ *XP*} is the set of input events, where *IPorts* represents the set of input ports and *XP* represents the set of values for the input ports
* *Y* = {(*p,v*)|*p* ∈ *OPorts*, *v* ∈ *YP* } is the set of output events, where *OPorts* represents the set of input ports and *YP* represents the set of values for the output ports
* *D* is the set of the component names and for each *d* ∈ *D*
* *EIC* is the set of external input couplings, *EIC* ∈ {((*Self*, *inSelf*), (*j*, *inj*))| *inSelf* ∈ *IPorts, j* ∈ *D*, *inj* ∈ *Iportsj*}
* *EOC* is the set of external output couplings, *EOC* ∈ {((*i*, *outi*), (*Self*, *outSelf*))| *outSelf* ∈ *OPorts, i* ∈ *D*, *outi* ∈ *Oportsi*}
* *IC* is the set of internal couplings, *IC* ∈ {(*i*, *outi*), (*j*, *inj*) )|*i,j* ∈ *D*, *outi* ∈ *Oportsi*, *inj* ∈ *Iportsj*}
* *select* is the tiebreaker function, where *select* ⊆ *D* 🡪 *D*, such that, for any nonempty subset *E, select (E)* ∈ *E*.

Figure 2.2 shows an example of a DEVS coupled model with three subcomponents, A1–A3. These basic models are interconnected through the corresponding I/O ports presented in the figure. The models are connected to the external coupled model through the EIC and EOC connectors. Keep in mind that A1–A3 are basic models (i.e., they can be atomic or coupled components).

### ***2.2.2. CELL-DEVS***

Cell-DEVS was defined as a combination of CA and DEVS with explicit timing delays. The DEVS formalism provides a framework for the construction of modular hierarchical models, allowing for model reuse, and for reducing development time and testing. In DEVS, basic models (called atomic) are specified as black boxes, and several DEVS models can be integrated forming a hierarchical structural model (called coupled). DEVS not only proposes a framework for model construction but also defines an abstract simulation mechanism that is independent of the model itself. Cell-DEVS define a cell as a DEVS model and a cellular automaton as a coupled model and introduces an explicit timing mechanism for each cell.

Using Cell-DEVS has different advantages. First, I have asynchronous model execution, which, as shown in my work with Giambiasi,3,4 results in improved simulation execution times. Timing constructions permit defining complex conditions for the cells in a simple fashion, as shown in also early work done on the topic with Norbert.5,6 As DEVS models are closed under coupling, seamless integration with other types of models in different formalisms is possible. The independent simulation engines permit these models to be executed interchangeably in single-processor, parallel or real-time simulators.

Cell-DEVS allows building cellular models in which each cell holds a state variable and a computing apparatus. This function is in charge of updating the cell state according to a local rule, by using the present cell state and those in a finite set of nearby cells (called its neighborhood). Each cell is defined as a DEVS atomic model, and it can be later integrated into a coupled model representing the cell space. A Cell-DEVS atomic model is defined as:

*TDC* = < *X , Y , I , S, E, delay, d, δint, δext, τ, λ, ta* >

Where

* *X* ∈***R***is a set of external input events
* *Y* ∈***R***is a set of external output events
* *I* is the model’s modular interface; *I* = < *η* + *γ*, *PX*, *PY*> defines the model interface. Here, *η*, *γ* ∈ ***N*** are the number of inputs coming from the neighborhood size and other inputs, and, for *I* = *X* or *I* = *Y*, *PI* is a port definition (input or output, respectively), where *PI* = {(*NiI*, *TiI*) / ∀ *i* [1, η + γ], *NiI* ∈ [*Ii*, *Iη*] (port name), and *TiI* =binary (port type)}
* *S* is the state set, where S = {(*s, σ*queue) / s ∈ **R**, *σ*queue = {((*v*1, *σ*1 ),...,(*v*m, *σ*m))/*m ∈* ***N*** *∧ m* < ∞ ∧ ∀(*i* ∈ *N*, *I* ∈ [1,*m*]), *v*i ∈ **R** ∧ *σ*i ∈ ***R0+*** ∪ ∞}}; for transport delays.

The formal specification of a cell with inertial delays only changes the state definition:

*S* = {(*s, f, σ*)/s, *f* ∈ *R*, and *σ* ∈ ***R0+*** ∪ ∞}

Where

* *E* is the set of values for the input external events
* *E* ∈ ***Rη+γ***, is the set of the external events (from the neighborhoods and other inputs)
* delay is the type of delay: transport, internial
* *δ* ∈ ***R0+***is the delay value for the cell
* *δ*int: *S* 🡪 *S* is the internal transition function
* *δext: Q x X* 🡪 *S* is the external transition function, where Q is the state set defined by Q = { (s, e) / s ∈ S, and e ∈ [0, ta(s)]}
* *τ: E* 🡪 *S* is the local transition function
* *λ: S* 🡪 *Y* is the output function
* ta: S 🡪 ***R0+*** ∪ ∞, is the time advance function.

A cell uses a set of input values E to compute its future state, which is obtained by applying the local computation function τ. A delay function is associated with each cell, deferring the output of the new state to the neighbor cells. Two types of delays were defined originally, which are used in most applications: inertial and transport delays. (When a transport delay is used, the future value will be added to a queue sorted by output time. Therefore, all the previous values that were scheduled for output but that have not yet been transmitted; will be kept in the queue. On the other hand, inertial delays use a preemptive policy: any previously scheduled output value, unless the same as the new computed one, will be deleted and the new one will be scheduled. This activation of the local computation is carried by the *δext* function.

After the basic behavior of a cell is defined, the complete cell space will be constructed by building a coupled Cell-DEVS model:

*GCC* = < *Xlist, Ylist, I, X , Y , n,* { *t1, tη* }*, N, C, B, Z* >

Where

* *Ylist* = {(*k,l*) / *k* ∈ [0,*m*], *l* ∈ [0,*n*]} is the list of output coupling
* *Xlist* = {(k,*l*) / k ∈ [0,m], *l* ∈ [0,*n*]} is the list of input coupling
* *I* = < *η, γx , γy , px , py* > represents the definition of the interface for the modular model whose size is *η* ∈ ***N***, *η* < ∞, *px* is the set of all input ports (*η* neigh-bor ports + *γy* external ports) and *py* is the set of all output ports (*η* neighbor ports + *γy* external ports)
* *X* ∈ ***R*** is a set of external input events
* *Y* ∈ ***R*** is a set of external output events
* *n* ∈ ***N*** is the dimension of the cell space
* {*t1*,…,*tη*} with *ti, i* ∈ ***N***, ∀*i* ∈ [1,*n*], is the number of cells in each of the dimensions
* *N* = {(*i, j*) / *i, j* ∈***Z*** *i, j* < ∞} is the neighborhood set
* *C* = { *Ck1*,...,*Ckn*)/ *ki* ∈ [0, *ti*]} is the cell space
* *B* ⊆ *C* ∪ {Ø} is the set of border cells
* *Z*: *PijYq* 🡪 *Pk1Xq*, where *PijYq* ∈ *Iij*, *PijXq* ∈ *Ik1*, *q* ∈ [0, *η*] and ∀(*f, g*)∈ *N*, *k* = (*i + f*) mod *m*; *l*=(*j + g*) mod *n*
* *Zij*: *Y(f,g)i* 🡪 *X(k,l)j* ∀ (f,g) *Ylisti*, and *(k,l) Xlistj*
* *select*, is the tie-breaking selector function, with the restriction that *select* ⊆ *m x n*  🡪 ∀ *E* ≠ {Ø}, *select(E)* ∈ *E.*

This specification defines a coupled model composed of an array of atomic cells. Each cell is connected to the cells defined in the neighborhood, but as the cell space is finite, either the borders are provided with a different neighborhood than the rest of the space, or they are ‘‘wrapped’’ meaning that cells in one border are connected with those in the opposite one. Finally, the *Z* function defines the internal and external coupling of cells in the model. This function translates the outputs of the *m*th output port in cell *Cij* into values for the *m*th input port of cell *Ckl*. Each output port will correspond to one neighbor and each input port will be associated with one cell in the inverse neighborhood.

The transport delay model allows introducing a delay between the occurrence of an external transition function and the state change of the cell. Only when the transport delay is consumed, the internal transition function is executed and the system changes its state. The *σ*queue is introduced because new external events can occur while the transport delay is consumed. These must be recorded and later executed by the internal transition functions.

In this definition, presented in figure 2.3, insert, first, tail and empty are the traditional functions employed to manage a FIFO queue. The external transition function schedules a new time for an internal transition function. To do so, it uses the value of the transport delay. The local transition function is executed using the new input values stored in E. Then, I update the times of the waiting events in the transport delay queue and update the state of the cell. This is only done only if there was a change; otherwise, the cell passivates. The output function is activated the next time for an internal event, and it generates an output based on the first element in the queue. The internal transition function then cleans up the queue and schedules the next internal event based on the queue information.

Text

Description automatically generated

Figure 2.3: Definition of transition functions for cells with transport delays [21].

The inertial delay constructions allow a behavior to be represented with a preemptive semantic. The construction says that, if an input value is not kept for a certain period (the inertial delay), the state change is not recorded. Instead, if the value is kept during that time, the state changes after the delay. To model this kind of delay, the transition functions are different.

The last arrived event can be preempted if a new external event (with a different value) arrives before the end of the inertial delay. If a new external event has the same value as the old one, the result is equivalent to having a unique external event. This is described in figure 2.4. If an event occurs in one cell, the neighbors are influenced through the execution of the *Z* function. Besides, certain cells in the space can be chosen as input and output cells, and they will be included in the *Xlist* and the *Ylist*, respectively. Xlist is a list of cell’s positions where the inputs to the model are received. *Ylist* records the cells whose outputs will be sent to the other models in the hierarchy.

When a Cell-DEVS model is executed, the *Zij* function translates inputs into outputs by using both lists. The names of the input and output ports are also defined by using the contents of the *Xlist* and *Ylist*.

The specification models here presented are independent of the simulation technique used. Therefore, they allow specifying the system behavior independently of the implementation details of the chosen simulation technique.

Text

Description automatically generated

Figure 2.4: External and internal transition function for inertial delays models [22].

If an event occurs in one cell, the neighbors are influenced through the execution of the *Z* function. Besides, certain cells in the space can be chosen as input and output cells, and they will be included in the *Xlis*t and the *Ylist*, respectively. *Xlist* is a list of cell’s positions where the inputs to the model are received. *Ylist* records the cells whose outputs will be sent to the other models in the hierarchy.

When a Cell-DEVS model is executed, the *Zij* function translates inputs into outputs by using both lists. The names of the input and output ports are also defined by using the contents of the *Xlist* and *Ylist*.

The specification models here presented are independent of the simulation technique used. Therefore, they allow specifying the system behavior independently of the implementation details of the chosen simulation technique.

### ***2.2.3. CD++ framework***

CD ++ is a tool built to implement DEVS and Cell-DEVS models. There are numerous tools, and some recent efforts include DesignDEV, DEVS-SOA, PythonDEVS, and others. The tool allows models to be defined according to the specifications introduced in the previous section. DEVS atomic models can be incorporated into a class hierarchy in C++. Coupled and Cell-DEVS models are defined using a specification language specially defined for this purpose, following DEVS and Cell-DEVS formal definitions. The tool includes an interpreter for a specification language that allows describing the behavior of each cell, including the local computing function and delay. In addition, it allows for defining the coupled model, including the size of the cell space and its connection with other DEVS models, the border and the initial state of each cell.

The behavior specification of a cell is defined using a set of rules, each indicating the future value for the cell’s state if a precondition is satisfied. The local computing function evaluates the first rule, and if the precondition does not hold, the following rules are evaluated until one of them is satisfied or there are no more rules. Figure 2.5 shows an example of the specification of a Cell-DEVS model developed using CD ++. The specification follows the Cell-DEVS coupled model’s formal definitions introduced in the previous section. In this case, Xlist = Ylist = {Ø}. Here, the dimension n = 2, therefore the set {*t1*, *t2*} is defined by the keywords width-height, which specifies the size of a two-dimensional (2D) cell space (in this example, *t1* = 20, *t2* = 40). The N set is defined by the sentence neighbors. The border (B) is wrapped. Using this information, CD ++ builds an executable cell space, defines the I/O ports, and the *Z* translation function following Cell-DEVS specifications.

Graphical user interface, text

Description automatically generated

Figure 2.5: A Cell-DEVS specification in CD++ [23].

The behavior of these rules, which define the local transition function, is defined using a set of rules in which there is a precondition to the right, a postcondition to the left, and a delay value between them. When the precondition is satisfied, the new value of the cell should change the postcondition value. The output of such value should be delayed using a transport, inertial, or another delay for the specified time. The tool’s main operators available to define rules include Boolean, comparison, arithmetic, neighborhood values, time, conditionals, angle conversion, pseudo-random numbers, error rounding and constants (i.e., gravitation, acceleration, light, Planck, etc.). In the example, the local computing function executes very simple rules. The first one indicates that whenever a cell state is 1 and the sum of the state values received in the input set E is 7, the cell state changes to 0. This state change will be spread to the neighboring cells after 200 ms. The second rule states that, whenever a cell state is 0 and the sum of the inputs whose value is 0 is smaller than 4, the cell value changes to 1 and the output is sent after 300 ms. In any other case (t = true), the result remains unchanged, and it will be spread to the neighbors after 110 ms. As I can see, cells evolve using a discrete-event approach.

The local computing function scans the specification, verifying the logical expressions included and computing the new state value for the cell. Several errors in the specification can be found at runtime, allowing the detection of inconsistencies in the model definition:

* Ambiguous models: a cell with the same precondition can produce different results.
* Incomplete models: no result exists for a certain precondition.
* Non-deterministic models: different preconditions are satisfied simultaneously. If they produce the same result, the simulation can continue, but the modeler is notified. Instead, if different results are found, the simulation should stop because the future state of the cell cannot be determined.

CD ++ extended the concept of ‘‘one state variable per cell’’ defined by CA and also includes the means to define ports to/from the neighbors, which are used to send/receive values from one cell to another in a coupled cell model. Using this idea, I need to declare state variables, which is done as follows (once declared, the state variables can be referenced in the rules). The first line declares the list of state variables that can be used by every cell. The second line declares the default initial values for these state variables:

StateVariables: pend temp vol

StateValues: 3.4 22 -5.2

The basic grammar for the rule is as follows:

[ < port\_assigns > ] < value >

[ < assignments >] < delay >

< precondition >

The precondition is a set of expressions that, if met, will result in the postcondition. This is a mix of three components: assigning values to output ports (optional, if there are any), to state variables (if any), and changing the value of the current cell’s main state variable. A variable is referenced by the name declared in the StateVariables sentence, preceded by a $, from any part of a rule, for instance:

rule: { (0,0,0) + 1 } { $temp:=$vol/2;

$pend:=(0,1,0); } 10 { (0,1,0) . 5.5 }

In the example, I am not using the optional port assignment section. Here, if the condition (0,1,0) . 5.5 is true, the variable temp will be assigned half of the vol value, pend will be assigned the value of the neighbor cell (0,1,0), and vol’s value will remain unmodified. The new value will be the one the cell holds plus one, and this value will be transmitted after 10 time units. The identifier ‘:=‘ is used to assign values to a state variable. Assignments can be placed in an expression within the rules (enclosed between curly brackets). A list of assignments can be defined, separated by semicolons.

I can use multiple I/O ports to communicate with the neighbors (besides a default port that transmits the cell’s value). They are defined as a list of neighbor port names as follows:

NeighborPorts: alarm weight number

The input and output neighbor ports share names, making it possible to calculate automatically the influences: an output port from a cell will influence exclusively the input. port with the same name in every cell in its neighborhood. In the example, I define three ports (alarm, weight, and number). When a cell outputs a value through one of these ports, it will be received by all its neighbor cells through their input ports with the same name. A cell can read the value sent by one of its neighbors, specifying the input port. Both the cell and port must be specified, separated by a tilde (~):

rule : 2 300 {(1,0)~weight . 20 }

In this case, if the cell receives an input in the weight port from the cell to the left and that value is larger than 20, the cell state will change to 2, and this change will be transmitted through the default output port 300 time units after that. As one might need to output values through many ports at the same time, the assignment can be used as many times as needed (each followed by a semicolon), as follows:

rule: { ~alarm := 1; ~weight := (0,-1)~weight} 100 { (0,1)~number . 50 }

In this example, if I receive a value larger than 50 from the port number in the cell to the right, I will wait 100 time units, and I will generate an output of 1 in the alarm port, and I will copy the weight value received from the cell to the left into the weight output port.

The rules defining the model’s coupling and those related to the behavior of a cell should be translated into an executable definition. To do so, the rule’s specifications are associated with a function’s identifier, which is registered by each cell, and each one of the rules is represented with a tuple (value, delay, condition) represented by a tree. To evaluate a rule, I evaluate the tree that represents the condition recursively. If the result of the evaluation is True, I evaluate the trees corresponding to the value and the delay, and the result of these evaluations are the values used by the cell. To do so, I built a lexical analyzer for the new language, whose grammar can be found in the Appendix. The < port\_assignments > produces a sequence of output operations triggered by the output function:

rule: { ~alarm := 0;send(alert, 1);} 100

{ portref(alert)=0 and ~alarm!= 0 }

## **2.3. Literature review**

### ***2.3.1. Simulation approaches in virology***

Modeling and simulation play an important role in virology, allowing us to study viruses and their components. Modeling methods commonly found in this field to date include mathematical and statistical models. Here are some viruses infection models observed and developed based on this simulation approach: the infection dynamics of Ebola [25], Hepatitis C [26], influenza A [27], HIV-AIDS [28] and Zika viruses [29]. These models are built as a system of ordinary differential equations that requires optimization [30]. However, in such mathematical models, the number of parameters is so large that a huge amount of computing power is needed to implement them. Therefore, it is very difficult to execute and find the optimal value for each parameter through lab experiments. Instead using the devs model will give better real-world understanding especially to simulate the dynamic behavior of viruses.

### ***2.3.2. DEVS and cell DEVS simulation approaches in virology***

As discussed earlier, DEVS takes a highly practical approach with small, highly coupled parameter sets and their behaviors already defined, then simulating dynamic virus behaviors becomes easier. This approach has been applied in simulating the replication of Covid-19 [31]. Especially, Cell-DEVS was defined as a combination of CA and DEVS that will help to solve the behavior of the individuals and the spatial dimension models. This is the case for the model of the spread of COVID-19 [32].

### ***2.3.3. Discussion (limits of previous approaches)***

To summarize, the simulation approaches described above all have their advantages and disadvantages, for mathematical model approach, it is a very useful theoretical model, but when applied in practice, a lot of difficulties and it is almost impossible to come up with an analytical solution. So to be more practical, applying the DEVS and CELL-DEVS approaches will bring high efficiency, save time and cost. However, having few parameters will lead to this approach not being highly general, which can lead to bias in the proposed hypothesis. In addition, no studies have applied this approach to simulate virus spread in lung tissue. This is the purpose of this project as an extension of the Covid-19 virus replication research.

# **CHAPTER 3 - THE PROPOSED METHOD AND ITS IMPLEMENTATION**

## **3.1. Overview of the proposed approach**

The approach I recommend is to use CELL-DEVS, a combination of cellular automation and discrete event simulation, to effectively characterize the tissue cell damage in the lung caused by Covid-19 brings. The proposed simulation method creates multi-layer models, they are closely related to each other based on the studied biological knowledge. These multilayer models include epithelial cell status, immune cell status, cytokine and virus concentrations. Combining these models allows biologists to observe the spread of the virus in lung tissue, the destruction of epithelial cells, and the body's response to this spread. From there, create scenarios as well as adjust input parameters to predict how much tissue damage the viruses can cause.

## **3.2. Cell-DEVS modeling of epithelial cell’s state**

### ***3.2.1. Definition***

The cell’s state can take five possible values: healthy(1), infected(2), secreting(3), non-secreting(4) and dead(5) as shown in the figure below:

Not ensured the viral replication

Do nothing

Infected by the virus

*Figure STYLEREF 1 \s 3. SEQ Figure \\* ARABIC \s 1 1: Epithelial cell’s state in Cell-Devs model.*

Ensured the viral replication

Released virions

[Figure 3.1: Epithelial cell’s state in CELL-DEVS model.](about:blank)

Healthy cells are maybe infected by external virus(es) (external event for this model). It can be infected by other virions from an infected cell(s) releasing virions). Once infected, the state changes to infected (state 2). If it is ensured the process of viral replication, its state will change to secreting(3), if it is not, it will be back to a healthy state. After releasing the virions, the cell turns its state to the dead(4).

### ***3.2.2. Rules***

I define the behavior of this model using the following set of rules:

* The initial rule describes the changes of the ports for which I want to change the value from the initial value -1.

rule :

**{*~initial* := 0; *~state* := *$s*; *~virion* := *$v*; *~uptake\_rate* := *$ur*; *~immune\_signal* := *$is*;}//Output**: these parameters define the output neighbor ports including initial (*the* port shows us whether the ports have been changed from the initial value -1 or not. Where 0 is the changed value and -1 is the unmodified value), state (state of the epithelial cell) will be released as state variable *$s* after a delay time, virion (the numbers of virion released) correspond to state variable *$v*, similar to uptake\_rate (percentage of virus absorbed on the surface of the cell) correspond to *$ur*, immune\_signal ( the intensity of the cell's immune signaling) correspond to *$is*.

**{*$ma* := round(uniform(1,100));}//Postcondition**: these parameters defines state variable *$ma* is the random number of virions for each cell from 1 to 100, if cell’s state become screting.

**1 //Delay: this parameter defines** the delay time will be 1 ms

**{(0,0,0)~initial = -1}//Precondition**: these parameters defines if their output initial port is -1  the rule will be executed.

When I start the simulation, the port initial values will always be -1, I will revalue the neighbor ports to correspond to the initial state. After that, change the value initial port to 0. And the rule is only executed the first time I simulate.

* The cell's healthy state rule describes the changes in the cell's state from a healthy state (1) to the infected state (2) or not.

rule :

**{*~state* := *$s*; *~uptake\_rate* := *$ur*; *~immune\_signal* := *$is*;}//Output**: these parameters defines the output neighbor ports include state (state of epithelial cell) will be released as state variable *$s* after a delay time, similar to uptake\_rate (percentage of virus absorbed on the surface of the cell) correspond to *$ur*, immune\_signal ( the intensity of the cell's immune signaling) correspond to *$is*.

**{*$s* := if(random<0.5\*(*(0,0,-1)~virus*/100)+0.5\*(1- *(0,0,1)~cytokine\_secreting*/1000) and *(0,0,-1)~virus* > 0, 2, 1);**

***$ur* := if(*$s* = 1, 0.1, 0);**

***$is* := if(*$s* = 1, 0, 1);}//Postcondition**: these parameters defines the state variable $s will depend on the probability of the number of viruses remaining on the cell surface dividing by 100 (the maximum number each infected cell can secrete) plus the compensating probability of the number of cytokines attached to the cell surface divided by 1000 (the maximum number of cytokines that can bind to the cell). If the random function is in the range of the probability and the presence of a virus on its cell membrane, then the $s will change to 2 ( infected). If not, it remains to 1 (healthy).

The state variable *$ur*  will change to 0.1 if *$s* is 1 or 0 if *$s* is not equal to 1.

The state variable *$is* will change to 0 if *$s* is 1 or 0 if *$s* is not equal to 1.

That means if the cell’s state is healthy, it absorbs 10 percent of viruses on its surface, and it doesn't secret immune signals. If the cell’s state is infected, it doesn’t absorb the virus on its surface, and its secrets immune signal has a value of 1.

**2000 //Delay: this parameter defines** the delay time will be 2000 ms

**{*(0,0,0)~state* = 1}//Precondition**: these parameters define if their output state port is 1 ( healthy) the rule will be executed.

* The cell's infected state rule describes the changes of the cell's state from the infected state (2) to the secreting state (3) or non-secreting state (4).

rule :

**{*~state* := *$s*; *~virion*:= *$v*;}//Output**: these parameters defines the output neighbor ports include state (state of epithelial cell) will be released as state variable *$s* after a delay time, virion (the numbers of virion released) correspond to state variable *$v*.

**{*$s* := if(random > (0,0,1)~cytokine\_secreting/1000, 3, 4);  *$v* := if( *$s* = 3, *$ma* ,0 );}//Postcondition** : these parameters defines the state variable $s will depend on the probability of the number of cytokine\_secreting divide by 1000 ( the maximum number of cytokines on the membrane of the cell. If the random function is out of the range of the probability, then the $s will change to 3 ( secreting) and it will secret random from 1 to 100 virions. If not, its change to 4 (non-secreting) .

**1000//Delay: this parameter defines** the delay time will be 1000 ms

**{*(0,0,0)~state* = 2}//Precondition**: these parameters defines if their output state port is 2 (infected) the rule will be executed.

This rule decides if the number of cytokines on the membrane of the infected cell is big enough, there will be a large percentage that the cell will not produce virions.

* The cell's secreting state rule describes the changes in the cell's state from the infected state (3) to the dead state (0).

rule :

**{*~state* := *$s*;  *~virion*:= *$v*; *~uptake\_rate* := *$ur*; *~immune\_signal* := 0;}**

**//Output**: these parameters define the output neighbor ports including state (state of the epithelial cell) will be released as state variable *$s* after a delay time, virion (the numbers of virion released) correspond to state variable *$v,* uptake\_rate (percentage of virus absorbed on the surface of the cell) correspond to *$ur*, immune\_signal ( the intensity of the cell's immune signaling) will be zero cause dead cell cant release immune signal.

**{*$s* := 0; *$v* := 0; *$ur* := 0.2;}//Postcondition**: these parameters defines the state variable *$s* will change to 0 (dead), *$v* (number of virions will be released is 0), the percentage of virus absorbed on the surface of the cell will be 0.2.

**4000//Delay: this parameter defines** the delay time will be 4000 ms.

**{*(0,0,0)~state* = 3}//Precondition**: these parameters defines if their output state port is 3 (secreting) the rule will be executed.

When the cell is dead, the viruses on the cell’s membrane will decrease slowly, and it will not secrete virions and immune signals anymore. After 4000ms the cell’s state will change from 3 (secreting) to 0 (dead)

* The cell's secreting state rule describes the changes in the cell's state from the non-secreting state (4) to a dead state (0).

rule :

**{*~state* := *$s*; ~immune\_signal := 0;}// Output**: these parameters defines the output neighbor ports include state (state of epithelial cell) will be released as state variable $s after a delay time, immune\_signal ( the intensity of the cell's immune signaling) will be zero cause dead cell cannot release immune signal.

**{*$s* := 0;}//Postcondition:**these parameters defines the state variable $s will change to 0 (dead).

**10000//Delay: this parameter defines** the delay time will be 10000 ms.

**{*(0,0,0)~state* = 4}//Precondition**: these parameters defines if their output state port is 4 (non-secreting) the rule will be executed.

When the cell is dead, the viruses on their cell’s membrane will decrease slowly , and it will not secrete immune signals. After, 10000ms the cell’s state will change from 4 (non-secreting) to 5 (dead).

### ***3.2.3. Measurement***

In parallel with the modeling, the statistical state of the epithelial cells is also very important in the evaluation of the results. Whenever there is a change in the state of any tissue cells, it will re-measure and make statistics at the time of the change.

## **3.3.** **Cell-DEVS modeling of immune cell’s state**

### ***3.3.1. Definition***

The immune cell’s state will initially always remain inactivated. When there is a large enough accumulation of cytokines, it will activate and inactivate immune cells to an active state, they will secrete cytokines into the extracellular field to help epithelial cells. In addition, epithelial cells whether in the active state or not will absorb a certain amount of cytokines on their surface.

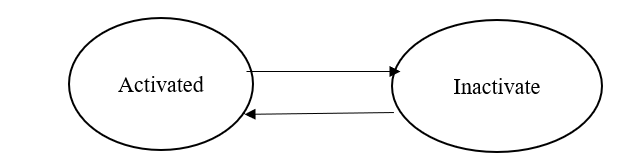


Figure 3.2: Immune cell’s state in CELL-DEVS model.

### ***3.3.2. Rules***

I define the behavior of this model using the following set of rules:

* The initial rule describes the changes of the ports for which I want to change the value from the initial value -1.

rule :

**{*~initial* := 0; *~state* := 0; *~uptake\_rate* := 0.1; *~immune\_signal* := *$is*;}//Output**: these parameters define the output neighbor ports including the initial ( the port shows us whether the ports have been changed from the initial value -1 or not. Where 0 is the changed value and -1 is the unmodified value), state (state of immune cell) will be 0 (inactive) at first, uptake\_rate (percentage of cytokine absorbed on the surface of the cell) will be 0.1 mean that all inactive immune cell will absorb 10 percent of cytokine in their membrane, immune\_signal ( the intensity of the cell's immune signaling) correspond to *$is*.

**1 //Delay: this parameter defines** the delay time will be 1 ms

**{(*0,0,0)~initial* = -1}//Precondition**:these parameters defines if their output initial port is -1  the rule will be executed.

When I start the simulation, the port initial values will always be -1, I will revalue the neighbor ports to correspond to the initial state. After that, change the value initial port to 0. And the rule is only executed the first time I simulate.

* The inactive immune cell rule describes the changes in the cell's state from inactive (0) to active (1).

rule :

**{*~state* := *$s*; *~immune\_signal* := *$is*;}//Output**: these parameters defines the output neighbor ports include state (state of immune cell) will be released as state variable $s after a delay time, immune\_signal ( the intensity of the cell's immune signaling) correspond to $is.

**{*$is* := if(*$s* = 1, 0.5, 0);}//Postcondition**: these parameters defines the state variable *$s* will depend on the probability of the number of cytokine on the cell surface dividing by 3000 (the maximum number of cytokine on the cell’s surface). If the random function is in the range of the probability and the number of cytokines on the cell surface is large enough to over 1000 cytokines, then the *$s* will change to 1(active). If not, it remains at 0 (inactive).

**1000//Delay: this parameter defines**s the delay time will be 1000 ms.

**{*(0,0,0)~state* = 0}//Precondition**:these parameters defines if their output state port is 0 (inactive) the rule will be executed.

* The active immune cell rule describes the changes in the cell's state from active (1) back to inactive (0).

rule :

**{*~state* := *$s*;}//Output**: these parameters define the output neighbor ports including the state (state of immune cell) that will be released as state variable $s after a delay time

**{ *$s* := 0;}//Postcondition**: these parameters define the state variable $s will depend on the probability of the number of cytokines on the cell surface dividing by 3000 (the maximum number of cytokines on the cell’s surface). If the random function is in the range of the probability and the number of cytokines on the cell surface is large enough to over 1000 cytokines, then the $s will change to 1 (active). If not, it remains at 0 (inactive).

**4000//Delay: this parameter defines** the delay time will be 4000 ms.

**{*(0,0,0)~state* = 1}//Precondition**: these parameters defines if their output state port is 1 (inactive) the rule will be executed.

After an immune cell is activated, it will be back to an inactive state in 4000 ms.

### ***3.3.3. Measurement***

In parallel with the modeling, the statistical state of the immune cells is also very important in the evaluation of the results. Whenever there is a change in the state of any immune cells, it will re-measure and make statistics at the time of the change.

## **3.4.** **Cell-DEVS modeling of cytokine concentration**

### ***3.4.1. Definition***

Cytokines will be secreted by the body when receiving immune signals from virus-infected cells and immune cells in an active state.

Regarding the relationship between the epithelial cell state model the and cytokine model, infected epithelial cells will secrete immune signals, alerting the body to secrete cytokines to help fight virus invasion. From there, epithelial cells, except for the dead state, can absorb cytokines to fight invading viruses.

Regarding the relationship between the immune cell model and the cytokine model, immune cells will initially always remain in a inactivate state. When there is a large enough accumulation of cytokines, it will activate immune cells to an active state, they will secrete cytokines into the extracellular field to help epithelial cells. In addition, epithelial cells whether in the active state or not will absorb a certain amount of cytokines on their surface.

### ***3.4.2. Rules***

I define the behavior of this model using the following set of rules:

* The initial rule describes the changes of the ports for which I want to change the value from the initial value -1.

rule :

**{*~initial* := 0; *~cytokine\_secreting* := *$cs*; *~cytokine\_movement* := round(*$cs*\*0.9);}**

**//Output** : these parameters define the output neighbor ports including initial ( the port shows us whether the ports have been changed from the initial value -1 or not. Where 0 is the changed value and -1 is the unmodified value), cytokine\_secreting (the number of cytokines adheres to the surface of the epithelium cells and immune cells) correspond to *$cs*, cytokine\_movement ( the amount of cytokine migrate to the surrounding cell membranes at the fixed time) will be about 90 percent of the cytokines on the surface of the cell membrane. According to the rule, where cytokines appear, they will also move to neighboring cells at the beginning.

**1//Delay: this parameter defines** the delay time will be 1 ms

**{*(0,0,0)~initial* = -1}//Precondition:** these parameters defines if their output initial port is -1  the rule will be executed.

When I start the simulation, the port initial values will always be -1, I will revalue the neighbor porcorrespondond to the initial state. After that change the value initial port to 0. And the rule is only executed the first time I simulate.

* The propagation cytokine rule describes a change in the amount of cytokine in the cell membrane, some will penetrate the cell membrane of the epithelium cells and immune cells they adhere to and most of them will migrate to adjacent cell membranes.

rule :

**{*~cytokine\_secreting* := *$cs*; *~cytokine\_movement* := *$cm*;}//Output**: these parameters define the output neighbor ports including cytokine\_secreting (the number of cytokines adheres to the surface of the epithelium cells and immune cells) correspond to $cs, cytokine\_movement ( the amount of cytokine migrate to the surrounding cell membranes at the fixed time) correspond to $cm.

**{ *$cm* := if( round(*$cs*\*0.9) > 0, round(*$cs*\*0.9), 0);**

***$cr*:=round(((1,0,0)*~cytokine\_movement*+(-1,0,0)*~cytokine\_movement* (0,1,0)*~cytokine\_movement* + (0,-1,0)*~cytokine\_movement*)/4);**

***$cs* := *$cs* + round((0,0,-1)~immune\_signal +((0,0,1)~immune\_signal ))\*0.7\*1000 - $cm + $cr - $cs\*(*(0,0,1)~uptake\_rate* + if(*(0,0,-1)~state* > 0,0.1,0));}//Postcondition**: these parameters defines the state variable $cm (the amount of cytokine migrate to the surrounding cell membranes) will be about 90 percent of the cytokine on the surface of the cell membrane, and if the amount of cytokine is too small to migrate is less than 1 (rounded no more than 1, mean its equal 0), then they will stay in the membrane of the cell and wait for the larger number to migrate, the state variable $cr (the amount of cytokine that the cell receives by moving the cytokine from the surrounding cells) it will add all the *~cytokine\_movement* ports of the surrounding cells containing the amount of cytokine moving at a fixed time, then divide by 4 (because the amount of cytokine moves evenly to 4 surrounding cells), the state variable *$cs* (the amount of cytokine on the cell membrane) is calculated based on the amount of cytokine signal from infected cells and active immune cells secrete in convert to cytokine by secretion from the body (each signal is corresponds to 0.7\*1000 = 100 cytokines) minus the amount of cytokine absorbed plus the amount of cytokine received from the surrounding cells minus the amount of cytokine moving to the surrounding cells.

**250//Delay: this parameter defines** the delay time will be 250 ms.

**{*t*}//Precondition**:these parameters define they don’t have precondition.

### ***3.3.1. Quantification***

In addition to cytokines simulating the concentration in lung tissue, my work to quantify cytokines to compare with the duration of infection, the number of infected tissue cells, as well as the number of activated immune cells is important.

## **3.5. Cell-DEVS modeling of virus spread**

### ***3.5.1. Definition***

I assume that the viral field is the field containing the viruses, it is the extracellular environment, located on the epithelial cell membrane.

Regarding the relationship between the epithelial cell state model the and virus propagation model, when the epithelial cells are in the secreting state, they will export virions to the extracellular environment and these virions will become viruses and spread to the surface of the surrounding epithelial cells. Uninfected epithelial cells will absorb some virus attached to their surface and potentially become infected. Since the virus is spreading rapidly in the extracellular environment, there will be many cases where epidermal cells are infected while the surrounding epidermal cells remain uninfected.

### **3.5.2. Rules**

I define the behavior of this model using the following set of rules:

* The initial rule describes the changes of the ports for which I want to change the value from the initial value -1.

rule :

**{*~initial* := 0; *~virus*:= *$vi*; *~virus\_movement* := (round(*$vi*\*0.8));}//Output**: these parameters define the output neighbor ports including the initial ( the port shows us whether the ports have been changed from the initial value -1 or not. Where 0 is the changed value and -1 is the unmodified value), virus (the amount of virus adheres to the surface of the epithelium cell correspond to *$vi*, virus\_movement ( the amount of virus migrate to the surrounding cell membranes at the fixed time) will be about 80 percent of the virus on the surface of the cell membrane. According to the rule, where viruses appear, they will also spread to neighboring cells at the beginning.

**1 //Delay: this parameter defines** the delay time will be 1 ms.

**{*(0,0,0)~initial* = -1}** these parameters defines if their output initial port is -1  the rule will be executed.

When I start the simulation, the port initial values will alwbe ays -1, I will revalue the neighbor ports correspond to the initial state. After that, change the value initial port to 0. And the rule is only executed the first time I simulate.

* The propagation virus rule describes a change in the amount of virus secreted from the cell membrane, some will penetrate the cell membrane of the epithelium cells they adhere to and most of them will migrate to adjacent cell membranes.

rule :

**{*~virus* := *$vi*; *~virus\_movement* := *$vim*;}//Output**: these parameters define the output neighbor ports including virus (the amount of virus adheres to the surface of the epithelium cell corresponds to *$vi*, virus\_movement (the amount of virus migrate to the surrounding cell membranes at the fixed time) correspond to $vim.

**{ *$vim* := if(round(*$vi*\*0.8) > 0, round(*$vi*\*0.8), 0);**

**$vir :=round(((1,0,0)~virus\_movement + (-1,0,0)~virus\_movement**

**+ (0,1,0)~virus\_movement + (0,-1,0)~virus\_movement)/4);**

**$vi := max(0,*$vi* + round((0,0,1)~virion/4 - *$vi*\*(0,0,1)~uptake\_rate) - *$vim* + $vir);}//Postcondition**: these parameters defines the state variable $vim (the amount of virus migrate to the surrounding cell membranes) will be about 80 percent of the virus on the surface of the cell membrane, and if the amount of virus is too small to migrate is less than 1 (rounded no more than 1, mean its equal 0), then they will stay in the membrane of the cell and wait for the larger number to migrate, the state variable $vir (the amount of virus that the cell receives by moving the virus from the surrounding cells) it will add all the ~virus\_movement ports of the surrounding cells containing the amount of virus moving at a fixed time, then divide by 4 (because the amount of virus moves evenly to 4 surrounding cells), the state variable $vi (the amount of virus on the cell membrane) is calculated based on the amount of virus secreted minus the amount of virus absorbed plus the amount of virus received from the surrounding cells minus the amount of virus moving to the surrounding cells.

**250//Delay: this parameter defines** the delay time will be 250 ms.

**{t}//Postcondition**: these parameters define they don’t have precondition.

### **3.5.3. Quantification**

In addition to simulating the spread of the virus in lung tissue, it is important to quantify the viruses to compare with the duration of infection, the number of infected tissue cells, as well as the number of activated immune cells.

## **3.6. Combination of the previous cell DEVS models (epithelial cell’s state, immunity cell’s state, cytokine concentration, and virus propagation)**

After describing the models in detail, it is very important to combine them in logical relationships. To generalize them and create an overall CELL-DEVS coupled model that simulates the spread of viruses in lung cells.

After describing the models in detail, it is very important to combine them in logical relationships. To generalize them and create an overall CELL-DEVS coupled model that simulates the spread of viruses in lung cells.

Diagram, schematic

Description automatically generated

Figure 3.3: State diagram and interactions of epithelial cell, immune cell, virus, cytokine.

As I have seen in figure 3.3. The relationship between models can be divided into three main parts.

Part one is the interaction between the virus model and the epithelium cell’s state. When cells are in the uninfected state, they can absorb some of the viruses on the surface, or they are in the secreting state, they release some virions into the extracellular environment. Part two is the interaction between the cytokine model and the epithelial cell state, when cells are infected, secreting or non-secreting, they all send immune signals asking to send them several cytokines against viruses in cells. The body will produce cytokines and proceed to send them back. In addition, dead epithelial cells will also destroy cytokines on their surface.

The third part is the interaction between the cytokine model and the immune cell model, initially, the immune cells are always in an inactive state. When there is a large number of cytokines attached to the surface, it will enter the activated state. This state will stimulate the body to secrete cytokines and it will last for ten hours and then return to the inactive state. Any immune cell in any state will absorb a certain amount of cytokines on its surface. In summary, defining the four sub-models and combining them creates a general model that simulates virus spread in lung cells in practice.

# **CHAPTER 4 –** **RESULTS AND DISCUSSION**

## **4.1. Development environment**

### ***4.1.1. Tools requirement***

This project uses 2 main development tools, CD++ 3.0 to create simulations from models and Python 3 for statistical results to evaluate the performance of simulated scenarios. The models run in a Linux environment.

### ***4.1.2. Development procedure***

Firstly, I create folders according to predefined scenarios. The details of the structure are more visible as shown in the figure below:

Graphical user interface

Description automatically generated with medium confidence

Figure 4.1: Project’s folders organization structure.

The structure of each folder is the same, they will include files:

* model.ma is the file containing all model information such as parameter definitions, rules, model dimensions, etc.
* model.val: contains state variables‘s initial values of specified cells that I want to initialize separately.
* simulation.log is the simulated result log file after executing the model (model.ma file) with CD++. It will contain the timestamp with each port change of the cells in the model.
* run.sh is bash script file to run the simulation using CD++ and save the log.
* .pal files are the files used to set the color of the cell’s outports, based on the set of eigenvalues corresponding to the submodel in it.

After setting up the Model.ma file and the initial static values of the cells in the model stored in Model.val, I conduct the simulation by running the bash file (run.bash).

Text

Description automatically generated

Figure 4.2: Simulation by executing run.sh.

I get a log file containing simulation results.

A picture containing text

Description automatically generated

Figure 4.3: The log file.

I can now observe the spread of the virus in lung tissue in real-time through four simulated models and their correlation using the visualization feature on the Carleton University web app provided.

To observe the simulation result on the web I will need to upload 4 files following the required web file model.ma, model.val, simulation.log and the pal file that will correspond to the model I want to see.Graphical user interface

Description automatically generated

Figure 4.4: Visualization of the web app’s interface.

Although each model has many ports to observe, each model has only one port which is its characteristic, the remaining ports are used to exchange information in their relationships.

I built these 4 models on 2-D space, so when combined they will create a coupled model with 3-D space with 4 layers. The first layer is the virus spread model, the second layer is the epithelial cell’s state model, the third layer is the cytokine model and the last layer is the epithelial immune cell state.

To observe the virus model, I upload model.ma, model.val, simulation.log and virus.pal corresponding to the model, then click the load simulation button. Once loaded, it will show the available ports for each layer. I only choose the virus port of the first layer. Press the play button to see the model change over time.

Graphical user interface

Description automatically generated

Figure 4.5: Epithelial cell’s state on the web app.

Similarly, with the epithelial cell’s state model, I upload the corresponding epithelial.pal. Select the state port of the 2nd layer to observe.

A picture containing graphical user interface

Description automatically generated

Figure 4.6: Immune cell’s state simulation on the web app.

Similarly, with the cytokine model, I upload cytokine.pal. Select the cytokine\_secreting port of the 3rd.

Graphical user interface

Description automatically generated

Figure 4.7: Cytokine simulation on the web app.

Similarly, with the immune cell’s state model, I upload immune.pal. Select the state port of the 4th layer.

A picture containing graphical user interface

Description automatically generated

Figure 4.8: Immune Cell’s state simulation on the web app.

In addition, the statistics of characteristic quantities such as the number of viruses in lung cells, the number of infected cells, etc. during the simulation is very important.

I have enumerated those characteristic quantities from the log file obtained at the end of the simulation by writing a python script that outputs the statistics excel file.

Graphical user interface, application, table, Excel

Description automatically generated

Figure 4.9: Excel file statistics the number of characteristics.

From that file, I continue to use the python script to graph the statistical chance of the characteristics over time.

Graphical user interface, application

Description automatically generated

Figure 4.10: Characteristic quantification graphs.

## **4.2. Application of the proposed cell-DEVS models to different scenarios**

There are many possible scenarios. Here I am just trying to create 3 general scenarios suitable for observing and drawing general conclusions. These scenarios are all simulated on a cellular space shaped like a miniature 2-D lung with the size of about 50 x 50 cells. The modeling rules in each scenario are similar, differing slightly in the parameters and initial values ​​of the static variables of a few cells.

### ***4.2.1.*** ***Scenario 1***

The first scenario I want to simulate is a scenario with a weak immune system and low penetration. Specifically, only one epithelial cell is initially infected. In terms of parameters, the number of cytokines that the body secretes to resist the spread of the virus is low, the state of the epithelial cell is infected, the probability of it changing to secreting is much higher. This probability is determined by the threshold of cytokines adhering to the epithelial cell surface, so this threshold is lower in this scenario. The expected outcome in this scenario is that the virus will spread very rapidly throughout the lung tissue and destroy all tissue cells.

### ***4.2.2.*** ***Scenario 2***

The second scenario would be the scenario where the body has high immunity and low viral penetration like the previous scenario where only one cell is infected. parametrically contrary to scenario one, the amount of cytokines that the body secretes will be large, and the cytokine threshold on the surface of epithelial cells determines the probability that infected cells will enter a secreting state will be low. The simulation result of this scenario is that the virus will propagate very slowly with low density and the amount of epithelial cell destruction will be very small.

### ***4.2.3******. Scenario 3***

The last scenario is still with the body with high immunity like the second scenario, so the parameters will be the same as the above scenario but the amount of virus penetration this time will be large. Specifically, eighteen epithelial cells were initially infected. It is expected that the simulation result of this scenario is that the virus will spread very slowly with low-density. Despite the large initial virus penetration, they will be isolated and gradually reduced leading to the number of epithelial cells being destroyed. damage is small and probably equal to scenario two.

## **4.3. Results and discussion**

### ***4.2.1. Scenario 1***

Video Simulation details of infection progression in four models of viral spread in lung tissue are captured and easily visualized.

(Link: <https://www.youtube.com/watch?v=fyM8I3FNpVY>)

Graphical user interface, application

Description automatically generated with medium confidence

Figure 4.11: Simulation of the progression of infection in four models over time in scenario 1.

I can see in Figure 4.11. Initially, only one epithelial cell is infected, all immune cells are inactive, extracellular virus is present on the surface of the infected cell, in addition, there is a large number of cytokines produced by the body provided on this cell.

After about 1500 minutes, the virus had spread to the superior lobes of both lungs, the epithelial cells surrounding the main bronchi were completely dead and the cells were in the secreting state, forming a ring around the main bronchus, the distribution of activated immune cells is similar to secreting epithelial cells with a greater density. The reason for this distribution can be explained by the fact that the density of cytokines in the trachea, main bronchus up to the superior lobe was very high.

At 3000 minutes, this time, the virus has moved to the middle lobe, their density in this part is very high because the number of epithelial cells in secreting state is mostly concentrated in the middle lobe, the epithelial cell part from the middle lobe. trachea to the superior lobe died the distribution of activated immune cells is similar to secreting epithelial cells with a greater density and the density of cytokines in the main bronchus up to the middle lobe was very high.

At 4500 minutes, most of the epithelial cells are dead, the rest are infected with the virus and most are secreting concentrated in the inferior lobes. The virus has spread throughout the lung tissue, but it is only concentrated in the lower lobes, and the other parts are quite low. The cytokine concentration shifts from the superior lobes to the inferior lobes.

At 6000 minutes, all epithelial cells are dead, only one immune cell is active. The virus is present everywhere on lung tissue but at a very low density, similar to cytokine concentration but still quite concentrated in the inferior lobes.

Chart, line chart

Description automatically generated

Figure 4.12: Simulation of the progression of infection in four models over time in scenario 1.

From figure 4.12, it can be seen that in about 6000 minutes, all epithelial cells changed from uninfected state to dead, in the period from 2000-3000 minutes, epithelial cells were infected with viruses the most. Observing the two lines representing the number of cells in secreting and non-secreting states, it can be seen that most infected cells will not be able to resist the virus and produce virions for export.

Chart, line chart

Description automatically generated

Figure 4.13: Quantification for the active immune cell during simulation time in scenario 1.

In figure 4.13, from the initial time to the end at 6000 minutes, the number of immune cells increased sharply and reached a maximum at about 2200 minutes and then gradually decreased to zero. This may be explained because the activation of an immune cell is proportional to the number of cytokines on its surface.

Chart, line chart

Description automatically generated

Figure 4.14: Quantification for the cytokine during simulation time in scenario 1.

Chart, line chart

Description automatically generated

Figure 4.15: Quantification for the virus in the extracellular environment during simulation time in scenario 1.

I can see the similarity in the two figures 4.14 and 4.15, they differ only fundamentally in viral and cytokine amplitude. Although they are not directly related to each other, this close similarity can be explained by the fact that they both have a homologous relationship with epithelial cells in the secreting state. The amount of virus in the extracellular environment depends on the number of epithelial cells in the secreting state, and the extracellular cytokines on the number of cells in the infected, secreting and non-secreting states. However, most of the infected cells will switch to the secreting state very quickly, so it can be inferred that the number of cytokines depends largely on the number of epithelial cells in the secreting state.

### ***4.2.2. Scenario 2***

Video Simulation details of infection progression in four models of viral spread in lung tissue are captured and easily visualized.

(Link: <https://www.youtube.com/watch?v=hePF_NY9f5s>)

Graphical user interface, application

Description automatically generated

Figure 4.16: Simulation of the progression of infection in four models over time in scenario 2.

I can see this in Figure 4.16. Initially, only one epithelial cell is infected, all immune cells are inactive, extracellular virus is present on the surface of the infected cell, in addition, there is a large number of cytokines produced by the body provided on this cell.

After about 1500 minutes, the virus had spread to the main bronchi with low density, the epithelial cells surrounding the main bronchi were completely dead and only one epithelial cell was in a secretory state and a few in a non-secretory state, and the distribution of activated immune cells is similar to dead epithelial cells with a greater density. The reason for this distribution can be explained by the fact that the density of the main bronchus was very high.

At 3000 minutes, this time, the viruses don’t seem to be able to spread further, their density is low and the distribution pattern remains almost the same as it was 1500 minutes ago, the number of immune cells in the activated state is reduced and spread out slightly horizontally because the distribution of cytokines also decreases and spreads horizontally.

At 4500 minutes, most of the cells in the main bronchI am dead and the rest of the tissue is in an uninfected state. The amount of activated immune cells continued to decrease as the cytokine distribution remained decreased. The virus distribution remains the same as before.

At 6000 minutes, the virus-induced damage to cells has stopped, leaving most of the dead main bronchi epithelial cells uninfected. there are no longer any activated immune cells, the distribution and density of the viruses remain the same and the amount of cytokines continues to decline sharply.

Chart, line chart

Description automatically generated

Figure 4.17: Simulation of the progression of infection in four models over time in scenario 2.

From figure 4.17, it can be seen that in about 6000 minutes, only about 1000 epithelial cells died and that number did not change at 5000 min. Observing the two lines representing the number of cells in secreting and non-secreting states, I see that the probability that infected cells will switch to secretory and non-secretory states is about the same.

Chart, line chart

Description automatically generated

Figure 4.18: Quantification for the active immune cell during simulation time in scenario 2.

In figure 4.18, from the initial time to 1000 minutes, the number of immune cells increased sharply and reached a maximum of about 2200 minutes and then they continuously increase and decrease in value. At 4000 minutes gradually decreased to zero. This may be explained because the activation of an immune cell is proportional to the number of cytokines on its surface.

Chart, line chart

Description automatically generated

Figure 4.19: Quantification for the cytokine during simulation time in scenario 2.

Chart, line chart

Description automatically generated

Figure 4.20: Quantification for the virus in the extracellular environment during simulation time in scenario 2.

I can see the similarity in the two figures 4.19 and 4.20 from 0 min to 1500 mins, they differ only fundamentally in viral and cytokine amplitude. It is different from the previous scenario, the amount of cytokine now depends mainly on the number of epithelial cells in the secretory state but also the non-secreting state.

### ***4.2.3. Scenario 3***

Video Simulation details of infection progression in four models of viral spread in lung tissue are captured and easily visualized.

(Link: <https://www.youtube.com/watch?v=jfYlylgopFc>)

Graphical user interface, application

Description automatically generated.

Figure 4.21: Simulation of the progression of infection in four models over time in scenario 3.

I can see this in Figure 4.21. Initially, a lot of epithelial cells are infected, all immune cells are inactive, and extracellular virus is present on the surface of the infected cell, in addition, there is a large number of cytokines produced by the body provided on this cell.

After about 1500 minutes, the virus had spread to the main bronchi with low density, the epithelial cells surrounding the main bronchi were completely dead and only one epithelial cell was in a secretory state and a few in a non-secretory state, and the distribution of activated immune cells is similar to dead epithelial cells with a greater density. The reason for this distribution can be explained by the fact that the density of the main bronchus was very high.

At 3000 minutes, at this time, the viruses don’t seem to be able to spread further, their density is low and the distribution pattern remains almost the same as it was 1500 minutes ago, the number of immune cells in the activated state is reduced and spread out slightly horizontally because the distribution of cytokines also decreases and spreads horizontally.

At 4500 minutes, most of the cells in the main bronchI am dead, and the rest of the tissue is in an uninfected state. The amount of activated immune cells continued to decrease as the cytokine distribution remained decreased. The virus distribution remains the same as before.

At 6000 minutes, the virus-induced damage to cells has stopped, leaving most of the dead main bronchi epithelial cells uninfected. there are no longer any activated immune cells, the distribution and density of the viruses remain the same and the amount of cytokines continues to decline sharply.

Chart, line chart

Description automatically generated

Figure 4.22: Simulation of the progression of infection in four models over time in scenario 3.

From figure 4.22, it is almost the same as case 2, only the number of cells in the infected state is quite high at first but also decreases over time.

Chart, line chart, histogram

Description automatically generated

Figure 4.23: Quantification for the active immune cell during simulation time in scenario 3.

In figure 4.23, from the initial time to 900 minutes, the number of immune cells increased sharply and reached a maximum at about 1900 minutes and then they continuously increase and decrease in value. At 5700 minutes gradually decreased to zero. This may be explained because the activation of an immune cell is proportional to the number of cytokines on its surface.

Chart, histogram

Description automatically generated

Figure 4.24: Quantification for the cytokine during simulation time in scenario 3.

Chart

Description automatically generated

Figure 4.25: Quantification for the virus in the extracellular environment during simulation time in scenario 3.

I can see the similarity in the two figures 4.24 and 4.25 from 0 min to 1200 mins, they differ only fundamentally in viral and cytokine amplitude. It is different from the previous scenario, the amount of cytokine now depends mainly on the number of epithelial cells in the secretory state but also in the non-secreting state.

GENERAL CONCLUSION

**Achieved result**

In this thesis project, I simulated virus propagation in lung cells through four closely linked models.

Even though, it is not yet fully simulated all the links between models. I have successfully applied the CELL-DEVS model to simulate the effects of COVID-19 on the host lungs by CD++. That is really interesting. Therefore, although this is only a preliminary simulation model, I can continue to perfect it to have a more sophisticated model.

**Restriction**

In this project, I aim to simulate the spread of COVID-19 in lung tissue so that it is as close to reality as possible. However, at present, there is still no actual amount of data, but I only collect simulation results by the ODE model in another study to evaluate the results. Therefore, there is no solution to prove whether this model is similar to reality or not. It is based mainly on my hypotheses when analyzing previous studies on this issue.

**Development prospects**

There are quite a few development directions for this interesting project such as:

* Creating more scenarios to better study the effect of the parameters in the rules.
* Develop the 3-D model for more practical convenience.
* Better analyze parts of lung tissue, thereby creating more accurate sub-models for virus spread in the lungs.

# **REFERENCES**

[1] Cao Y-C, Deng Q-X, Dai S-X, “Remdesivir for severe acute respiratory syndrome coronavirus 2 causing COVID-19: An evaluation of the evidence”, 2020.

[2] Wang D, Yin Y, Hu C, Liu X, Zhang X, Zhou S, et al, “A systems approach to infectious disease”, 2020-2021.

[3] Eckhardt M, Hultquist JF, Kaake RM, Hüttenhain R, Krogan NJ, “Clinical course and outcome of 107 patients infected with the novel coronavirus, SARS-CoV-2, discharged from two hospitals in Wuhan, China”, 2020-2021.

[4] Dolley S, “Big Data’s Role in Precision Public Health. Front Public Health”, 2018

[5] Garira W, “A primer on multiscale modelling of infectious disease systems. Infectious Disease Modelling”, 2018, pp.176-191.

[6] Dee KU, Hammer DA, Shuler ML, “A model of the binding, entry, uncoating, and RNA synthesis of Semliki Forest virus in baby hamster kidney (BHK-21) cells. Biotechnology and Bioengineering”, 1995.

[7] English TJ, Hammer DA, “Brownian adhesive dynamics (BRAD) for simulating the receptor-mediated binding of viruses”, 2004.

[8] Boireau S, Maiuri P, Basyuk E, de la Mata M, Knezevich A, Pradet-Balade B, et al, “The transcriptional cycle of HIV-1 in real-time and live cells”, 2007.

[9] Roldão A, Vieira HLA, Charpilienne A, Poncet D, Roy P, Carrondo MJT, et al, “Modeling rotavirus-like particles production in a baculovirus expression vector system: Infection kinetics, baculovirus DNA replication, mRNA synthesis and protein production”, 2007.

[10] Purohit PK, Inamdar MM, Grayson PD, Squires TM, Kondev J, Phillips R, “Forces during bacteriophage DNA packaging and ejection”, 2004.

[11] Yang Q, Catalano CE, “A minimal kinetic model for a viral DNA packaging machine”, 2004.

[12] Graw F, Perelson A, “Spatial aspects of HIV infection”, 2013, pp.3–31.

[13] Wainer, G.A, “An Introduction to Cellular Automata Models with Cell-DEVS”, 2019, pp.1534–1548.

[14] Zeigler, B.P., Praehofer, H., Kim, T.G, “Theory of Modeling and Simulation. Integrating Discrete Event and Continuous Complex Dynamic Systems”, 2000.

[15] Shang, J., Y. Wan, C. Luo, G. Ye, Q. Geng, A. Auerbach, and F. Li, “Cell entry mechanisms of SARS-CoV-2”, 2020.

[16] Naqvi, A. A. T., K. Fatima, T. Mohammad, U. Fatima, I. K. Singh, A. Singh, S. M. Atif, G. Hariprasad, G. M.Hasan, and M. I. Hassan, “Insights into SARS-CoV-2 genome, structure, evolution, pathogenesis and therapies: Structural genomics approach”. Biochimica et Biophysica Acta (BBA)-Molecular Basis of Disease”, 2020.

[17] Philip V’kovski, Annika Kratzel, Silvio Steiner, Hanspeter Stalder and Volker Thiel, “ Coronavirus biology and replication: implications for SARS-CoV-2”, 3/2021, pp.156.

[18] T. J. Sego ,Josua O. Aponte-Serrano ,Juliano Ferrari Gianlupi,Samuel R. Heaps,Kira Breithaupt,Lutz Brusch,Jessica Crawshaw,James M. Osborne,Ellen M. Quardokus,Richard K. Plemper,James A. Glazier, “A modular framework for multiscale, multicellular, spatiotemporal modeling of acute primary viral infection and immune response in epithelial tissues and its application to drug therapy timing and effectiveness”, 11/2020.

[19-20] Gabriel A. Wainer, “Discrete-Event Modeling and Simulation A Practitioner’s Approach”, 2009, pp.37-38.

[21-23] Gabriel A. Wainer, “Advanced Cell-DEVS modeling applications: a legacy of Norbert Giambiasi”, 4/2018, pp.4-6.

[24] Hattaf, Khalid and Elaiw, Ahmed M and Lashari, Abid A and Yousfi, Noura, “Mathematical Modeling in Virology by Differential Equations”, 2018.

[25] Madelain, V., S. Baize, F. Jacquot, S. Reynard, A. Fizet, S. Barron, C. Solas, B. Lacarelle, C. Carbonnelle, F. Mentré et al, “Ebola viral dynamics in nonhuman primates provides insights into virus immuno-pathogenesis and antiviral strategies”, 2018.

[26] Guedj, J., H. Dahari, and A. S. Perelson, “Understanding the nature of early HCV RNA blips and the use of mathematical modeling of viral kinetics during IFN-based therapy”, 2011.

[27] Handel, A., L. E. Liao, and C. A. Beauchemin, “Modeling and Control for HIV/AIDS Transmission in China Based on Data from 2004 to 2016”, 2018.

[28] Li, Z., Z. Teng, and H. Miao, “Forces during bacteriophage DNA packaging and ejection”, 2017.

[29] Best, K., and A. S. Perelson, “Mathematical modeling of within-host Zika virus dynamics”, 2018.

[30] Quintela, B. d. M., J. M. Conway, J. M. Hyman, J. Guedj, R. W. Dos Santos, M. Lobosco, and A. S. Perelson, “A new age-structured multiscale model of the hepatitis C virus life-cycle during infection and therapy with direct-acting antiviral agents”, 2018.

[31] Ali Ayadi, Claudia Frydman, Wissame Laddada, Lina F. Soualmia, Cecilia Zanni-Merk, “Combining DEVS and semantic technologies for modeling the SARS-COV-2 replication machinery”, 2020.

[32] Román Cárdenas, Gabriel A. Wainer, Cristina Ruiz-Martin, “Cell-DEVS Models for the Spread of COVID-19”, 2021.

[33] Bernard P. Zeigler, et al, “Theory of Modeling and Simulation”, 2000.

[34] Bernard P. Zeigler, et al, “Theory of Modeling and Simulation”, 1976.

[35] Hans Vangheluwe, “DEVS as a Common Denominator for Multi-formalism Hybrid Systems Modelling”, 2000.

[36] Clarence Ellis, Karim Keddara and Jacques Wainer, “Modeling Workflow Dynamic Changes Using Timed Hybrid Flow Nets”, 1998.

[37] Bernard Phillip Zeigler, “Embedding DEV&DESS in DEVS”, 2006.

[38] Gabriel A. Wainer et al, “An Introduction to DEVS Standardization”, 2010.

[39] Carrasco Pro S, Dafonte Imedio A, Santoso CS, Gan KA, Sewell JA, Martinez M, et al, “Global landscape of mouse and human cytokine transcriptional regulation. Nucleic Acids Researchs”, 2018.

[40] Akira S, Uematsu S, Takeuchi O, “Pathogen Recognition and Innate Immunity Cell”, 2006.

[41] Ahlquist P, Noueiry AO, Lee W-M, Kushner DB, Dye BT, “Host Factors in Positive-Strand RNA Virus Genome Replication”, 2003.

# **APPENDIX**

**Source code: Model.ma**

[top]

components : cellular

[cellular]

type : cell

dim : (50, 50, 5)

delay : transport

defaultDelayTime : 100

border : wrapped

neighbors : cellular(-1,0,0) cellular(0,0,0)

neighbors : cellular(0,1,0) cellular(1,0,0) cellular(0,-1,0)

neighbors : cellular(0,0,1) cellular(0,0,-1)

initialvalue : -1

NeighborPorts: initial state virion virus virus\_movement uptake\_rate immune\_signal cytokine\_movement cytokine\_secreting

statevariables: s v immunity\_rate vi vim vir ur ma is cm cr cs

statevalues: 1 0 0 0 0 0 0.1 0 0 0 0 0

initialvariablesvalue: model.val

localtransition: local

zone : virus { (3,23,1)..(3,25,1) (4,23,1)..(4,25,1) (5,23,1)..(5,25,1) (6,23,1)..(6,25,1) (6,29,1)..(6,30,1) (7,16,1)..(7,21,1) (7,23,1)..(7,25,1) (7,27,1)..(7,32,1) (8,15,1)..(8,21,1) (8,23,1)..(8,25,1) (8,27,1)..(8,33,1) (9,14,1)..(9,21,1) (9,23,1)..(9,25,1) (9,27,1)..(9,34,1) (10,13,1)..(10,21,1) (10,23,1)..(10,25,1) (10,27,1)..(10,35,1) (11,12,1)..(11,21,1) (11,23,1)..(11,25,1) (11,27,1)..(11,36,1) (12,11,1)..(12,21,1) (12,23,1)..(12,25,1) (12,27,1)..(12,37,1) (13,10,1)..(13,38,1) (14,9,1)..(14,38,1) (15,9,1)..(15,39,1) (16,8,1)..(16,23,1) (16,24,1)..(16,39,1) (17,8,1)..(17,23,1) (17,25,1)..(17,40,1) (18,8,1)..(18,23,1) (18,25,1)..(18,40,1) (19,7,1)..(19,22,1) (19,25,1)..(19,40,1) (20,7,1)..(20,23,1) (20,25,1)..(20,40,1) (21,7,1)..(21,23,1) (21,25,1)..(21,41,1) (22,6,1)..(22,22,1) (22,25,1)..(22,41,1) (23,6,1)..(23,22,1) (23,26,1)..(23,41,1) (24,6,1)..(24,22,1) (24,26,1)..(24,41,1) (25,6,1)..(25,22,1) (25,26,1)..(25,41,1) (26,6,1)..(26,22,1) (26,26,1)..(26,41,1) (27,6,1)..(27,22,1) (27,26,1)..(27,41,1) (28,6,1)..(28,22,1) (28,26,1)..(28,41,1) (29,6,1)..(29,22,1) (29,26,1)..(29,41,1) (30,6,1)..(30,22,1) (30,26,1)..(30,41,1) (31,6,1)..(31,22,1) (31,26,1)..(31,41,1) (32,6,1)..(32,21,1) (32,26,1)..(32,41,1) (33,6,1)..(33,21,1) (33,27,1)..(33,41,1) (34,6,1)..(34,21,1) (34,27,1)..(34,41,1) (35,5,1)..(35,20,1) (35,29,1)..(35,41,1) (36,5,1)..(36,16,1) (36,30,1)..(36,41,1) (37,6,1)..(37,11,1) (37,32,1)..(37,40,1) (38,37,1)..(38,40,1) }

zone : epithelial-cell { (3,23,2)..(3,25,2) (4,23,2)..(4,25,2) (5,23,2)..(5,25,2) (6,23,2)..(6,25,2) (6,29,2)..(6,30,2) (7,16,2)..(7,21,2) (7,23,2)..(7,25,2) (7,27,2)..(7,32,2) (8,15,2)..(8,21,2) (8,23,2)..(8,25,2) (8,27,2)..(8,33,2) (9,14,2)..(9,21,2) (9,23,2)..(9,25,2) (9,27,2)..(9,34,2) (10,13,2)..(10,21,2) (10,23,2)..(10,25,2) (10,27,2)..(10,35,2) (11,12,2)..(11,21,2) (11,23,2)..(11,25,2) (11,27,2)..(11,36,2) (12,11,2)..(12,21,2) (12,23,2)..(12,25,2) (12,27,2)..(12,37,2) (13,10,2)..(13,38,2) (14,9,2)..(14,38,2) (15,9,2)..(15,39,2) (16,8,2)..(16,23,2) (16,24,2)..(16,39,2) (17,8,2)..(17,23,2) (17,25,2)..(17,40,2) (18,8,2)..(18,23,2) (18,25,2)..(18,40,2) (19,7,2)..(19,22,2) (19,25,2)..(19,40,2) (20,7,2)..(20,23,2) (20,25,2)..(20,40,2) (21,7,2)..(21,23,2) (21,25,2)..(21,41,2) (22,6,2)..(22,22,2) (22,25,2)..(22,41,2) (23,6,2)..(23,22,2) (23,26,2)..(23,41,2) (24,6,2)..(24,22,2) (24,26,2)..(24,41,2) (25,6,2)..(25,22,2) (25,26,2)..(25,41,2) (26,6,2)..(26,22,2) (26,26,2)..(26,41,2) (27,6,2)..(27,22,2) (27,26,2)..(27,41,2) (28,6,2)..(28,22,2) (28,26,2)..(28,41,2) (29,6,2)..(29,22,2) (29,26,2)..(29,41,2) (30,6,2)..(30,22,2) (30,26,2)..(30,41,2) (31,6,2)..(31,22,2) (31,26,2)..(31,41,2) (32,6,2)..(32,21,2) (32,26,2)..(32,41,2) (33,6,2)..(33,21,2) (33,27,2)..(33,41,2) (34,6,2)..(34,21,2) (34,27,2)..(34,41,2) (35,5,2)..(35,20,2) (35,29,2)..(35,41,2) (36,5,2)..(36,16,2) (36,30,2)..(36,41,2) (37,6,2)..(37,11,2) (37,32,2)..(37,40,2) (38,37,2)..(38,40,2) }

zone : cytokine { (3,23,3)..(3,25,3) (4,23,3)..(4,25,3) (5,23,3)..(5,25,3) (6,23,3)..(6,25,3) (6,29,3)..(6,30,3) (7,16,3)..(7,21,3) (7,23,3)..(7,25,3) (7,27,3)..(7,32,3) (8,15,3)..(8,21,3) (8,23,3)..(8,25,3) (8,27,3)..(8,33,3) (9,14,3)..(9,21,3) (9,23,3)..(9,25,3) (9,27,3)..(9,34,3) (10,13,3)..(10,21,3) (10,23,3)..(10,25,3) (10,27,3)..(10,35,3) (11,12,3)..(11,21,3) (11,23,3)..(11,25,3) (11,27,3)..(11,36,3) (12,11,3)..(12,21,3) (12,23,3)..(12,25,3) (12,27,3)..(12,37,3) (13,10,3)..(13,38,3) (14,9,3)..(14,38,3) (15,9,3)..(15,39,3) (16,8,3)..(16,23,3) (16,24,3)..(16,39,3) (17,8,3)..(17,23,3) (17,25,3)..(17,40,3) (18,8,3)..(18,23,3) (18,25,3)..(18,40,3) (19,7,3)..(19,22,3) (19,25,3)..(19,40,3) (20,7,3)..(20,23,3) (20,25,3)..(20,40,3) (21,7,3)..(21,23,3) (21,25,3)..(21,41,3) (22,6,3)..(22,22,3) (22,25,3)..(22,41,3) (23,6,3)..(23,22,3) (23,26,3)..(23,41,3) (24,6,3)..(24,22,3) (24,26,3)..(24,41,3) (25,6,3)..(25,22,3) (25,26,3)..(25,41,3) (26,6,3)..(26,22,3) (26,26,3)..(26,41,3) (27,6,3)..(27,22,3) (27,26,3)..(27,41,3) (28,6,3)..(28,22,3) (28,26,3)..(28,41,3) (29,6,3)..(29,22,3) (29,26,3)..(29,41,3) (30,6,3)..(30,22,3) (30,26,3)..(30,41,3) (31,6,3)..(31,22,3) (31,26,3)..(31,41,3) (32,6,3)..(32,21,3) (32,26,3)..(32,41,3) (33,6,3)..(33,21,3) (33,27,3)..(33,41,3) (34,6,3)..(34,21,3) (34,27,3)..(34,41,3) (35,5,3)..(35,20,3) (35,29,3)..(35,41,3) (36,5,3)..(36,16,3) (36,30,3)..(36,41,3) (37,6,3)..(37,11,3) (37,32,3)..(37,40,3) (38,37,3)..(38,40,3) }

zone : immune { (3,23,4)..(3,25,4) (4,23,4)..(4,25,4) (5,23,4)..(5,25,4) (6,23,4)..(6,25,4) (6,29,4)..(6,30,4) (7,16,4)..(7,21,4) (7,23,4)..(7,25,4) (7,27,4)..(7,32,4) (8,15,4)..(8,21,4) (8,23,4)..(8,25,4) (8,27,4)..(8,33,4) (9,14,4)..(9,21,4) (9,23,4)..(9,25,4) (9,27,4)..(9,34,4) (10,13,4)..(10,21,4) (10,23,4)..(10,25,4) (10,27,4)..(10,35,4) (11,12,4)..(11,21,4) (11,23,4)..(11,25,4) (11,27,4)..(11,36,4) (12,11,4)..(12,21,4) (12,23,4)..(12,25,4) (12,27,4)..(12,37,4) (13,10,4)..(13,38,4) (14,9,4)..(14,38,4) (15,9,4)..(15,39,4) (16,8,4)..(16,23,4) (16,24,4)..(16,39,4) (17,8,4)..(17,23,4) (17,25,4)..(17,40,4) (18,8,4)..(18,23,4) (18,25,4)..(18,40,4) (19,7,4)..(19,22,4) (19,25,4)..(19,40,4) (20,7,4)..(20,23,4) (20,25,4)..(20,40,4) (21,7,4)..(21,23,4) (21,25,4)..(21,41,4) (22,6,4)..(22,22,4) (22,25,4)..(22,41,4) (23,6,4)..(23,22,4) (23,26,4)..(23,41,4) (24,6,4)..(24,22,4) (24,26,4)..(24,41,4) (25,6,4)..(25,22,4) (25,26,4)..(25,41,4) (26,6,4)..(26,22,4) (26,26,4)..(26,41,4) (27,6,4)..(27,22,4) (27,26,4)..(27,41,4) (28,6,4)..(28,22,4) (28,26,4)..(28,41,4) (29,6,4)..(29,22,4) (29,26,4)..(29,41,4) (30,6,4)..(30,22,4) (30,26,4)..(30,41,4) (31,6,4)..(31,22,4) (31,26,4)..(31,41,4) (32,6,4)..(32,21,4) (32,26,4)..(32,41,4) (33,6,4)..(33,21,4) (33,27,4)..(33,41,4) (34,6,4)..(34,21,4) (34,27,4)..(34,41,4) (35,5,4)..(35,20,4) (35,29,4)..(35,41,4) (36,5,4)..(36,16,4) (36,30,4)..(36,41,4) (37,6,4)..(37,11,4) (37,32,4)..(37,40,4) (38,37,4)..(38,40,4) }

[epithelial-cell]

rule : {~initial := 0; ~state := $s; ~virion := $v; ~uptake\_rate := $ur; ~immune\_signal := $is;} { $immunity\_rate := uniform(0.0, 1); $ma := round(random\*100);}

1 {(0,0,0)~initial = -1}

rule : {~state := $s; ~uptake\_rate := $ur; ~immune\_signal := $is;}

{ $s := if( random < 0.5\*((0,0,-1)~virus/100) + 0.5\*(1 - (0,0,1)~cytokine\_secreting/1000)

and (0,0,-1)~virus > 0, 2, 1);

$ur := if($s = 1, 0.1, 0);

$is := if($s = 1, 0, 1);

}

37500

{ (0,0,0)~state = 1 }

rule : {~state := $s; ~virion:= $v; ~uptake\_rate := 0; }

{ $s := if(random > (0,0,1)~cytokine\_secreting/1000, 3, 4); $v := if($s = 3, $ma ,0 ); }

37500

{(0,0,0)~state = 2}

rule : { ~state := $s; ~immune\_signal := 0; } { $s := 0; } 150000 { (0,0,0)~state = 4 }

rule : { ~state := $s; ~virion:= $v; ~uptake\_rate := $ur; ~immune\_signal := 0;}

{ $s := 0; $v := 0; $ur := 0.2;}

600000

{ (0,0,0)~state = 3 }

rule : {~virion:= 0;} 150000 {t}

[virus]

rule : {~initial := 0; ~virus:= $vi; ~virus\_movement := (round($vi\*0.8));} 1 {(0,0,0)~initial = -1}

rule : { ~virus := $vi; ~virus\_movement := $vim;}

{ $vim := if(round($vi\*0.8) > 0, round($vi\*0.8), 0);

$vir := round(((1,0,0)~virus\_movement + (-1,0,0)~virus\_movement + (0,1,0)~virus\_movement + (0,-1,0)~virus\_movement)/4);

$vi := max(0,$vi + round((0,0,1)~virion/4 - $vi\*(0,0,1)~uptake\_rate) - $vim + $vir);

}

37500

{ t }

[cytokine]

rule : {~initial := 0; ~cytokine\_secreting := $cs; ~cytokine\_movement := round($cs\*0.9);} 1 {(0,0,0)~initial = -1}

rule : { ~cytokine\_secreting := $cs; ~cytokine\_movement := $cm; }

{

$cm := if( round($cs\*0.9) > 0, round($cs\*0.9), 0);

$cr := round(((1,0,0)~cytokine\_movement + (-1,0,0)~cytokine\_movement + (0,1,0)~cytokine\_movement + (0,-1,0)~cytokine\_movement)/4);

$cs := $cs + round((0,0,-1)~immune\_signal +((0,0,1)~immune\_signal ))\*1000 - $cm + $cr

- $cs\*((0,0,1)~uptake\_rate + if((0,0,-1)~state > 0,0.1,0));

}

37500

{t}

[immune]

rule : {~initial := 0; ~state := 0; ~uptake\_rate := 0.1; ~immune\_signal := $is;} 1 {(0,0,0)~initial = -1}

rule : {~state := $s; ~immune\_signal := $is;} { $s := if(random < (0,0,-1)~cytokine\_secreting/3000 and (0,0,-1)~cytokine\_secreting > 1000, 1, 0); $is := if($s = 1, 0.5, 0);} 150000 {(0,0,0)~state = 0}

rule : {~state := $s;} { $s := 0; } 600000 {(0,0,0)~state = 1}

[local]

rule : {~state := -1;} 150000 {t}