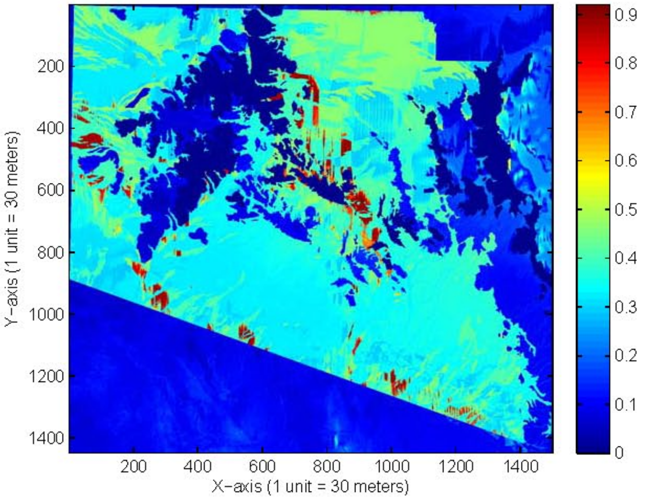
USGS
science for a changing world

Multilevel Methodology for Simulation of Spatio-Temporal Systems with Heterogeneous Activity; Application to Spread of Valley Fever Fungus

By Rajanikanth Jammalamadaka



A dissertation submitted to the faculty of the Department of Electrical and Computer Engineering, in partial fulfillment of the requirements for the degree of Doctor of Philosophy, Graduate College, The University of Arizona, 2008.

Open-File Report 2009-1241

U.S. Department of the Interior
U.S. Geological Survey

U.S. Department of the Interior
KEN SALAZAR, Secretary

U.S. Geological Survey
Marcia K. McNutt, Director

U.S. Geological Survey, Reston, Virginia 2009

For product and ordering information:
World Wide Web: http://www.usgs.gov/pubprod/
Telephone: 1-888-ASK-USGS

For more information on the USGS—the Federal source for science about the Earth,
its natural and living resources, natural hazards, and the environment:
World Wide Web: http://www.usgs.gov/
Telephone: 1-888-ASK-USGS

Fiscal support for the work was provided from the USGS Mineral Resources Program and a project grant from the National Park Service.

Suggested citation:
Jammalamadaka, Rajanikanth, 2009, Multilevel methodology for simulation of spatio-temporal systems with heterogeneous activity: application to spread of valley fever fungus: U.S. Geological Survey Open-File Report 2009-1241, 105 p. [http://pubs.usgs.gov/of/2009/1241/].

Any use of trade, product, or firm names is for descriptive purposes only and does not imply endorsement by the U.S. Government.

Cover: illustration showing the resulting ground model for favorableness of the ground for the growth of the valley fever fungus for the Organ Pipe National Monument in Southwestern Arizona (from figure 3.2 on page 33)

Multilevel Methodology for Simulation Of Spatio-Temporal Systems With Heterogeneous Activity: Application To Spread of Valley Fever Fungus

by

Rajanikanth Jammalamadaka

A Dissertation Submitted to the Faculty of the

Department Of Electrical and Computer Engineering

In Partial Fulfillment of the Requirements
For the Degree of

Doctor of Philosophy

In the Graduate College

The University of Arizona

2008

(Official approval page

from the Graduate College)

Statement by Author

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this dissertation are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

Signed: ______________________________

Acknowledgements

I would like to thank my parents and my brother for their continued love and support. I am what I am because of my parents.

I would like to thank my advisor Prof. Bernard P. Zeigler for his guidance, support and encouragement throughout this endeavour. His attention to detail and follow-up were instrumental to the successful completion of this dissertation.

I would like to thank Dr. Mark E. Gettings for his support and encouragement. He was very patient in answering all my questions and helped me a lot in finishing this dissertation.

I would like to thank Dr. Mark Bultman for answering my questions regarding the ground model.

I would like to thank Dr. Nathan Goodman and Dr. Hal Tharp for reviewing my dissertation and for serving on my dissertation defense committee.

I would like to thank the United States Geological Survey for funding my research.

I would like to thank Dr. Jim Nutaro for patiently answering many of my questions.

I would like to thank Prof. Leonid Kunyansky for helping me learn Numerical Linear Algebra. He was very patient in answering many of my questions while I was learning the subject.

I would like to thank Sri Swami Tattvavidananda Saraswati for his blessings and his lectures which gave me peace of mind in times of trouble.

I would like to thank my friends Ravi Parimi, Srinivas Singavarapu, Bhaskar, Vinod, Vaidehi for creating a friendly atmosphere and for making my stay in Tucson memorable.

I would like to thank my lab mates at ACIMS for creating a friendly and entertaining work environment.

Dedication

To my parents:

Sri. Srimannarayana Jammalamadaka,

Smt. Vijayalakshmi Jammalamadaka.

Table of Contents

Table of Contents—*Continued*

List of Tables

List of Figures

List of Figures—*Continued*

List of Figures—*Continued*

Abstract

Spatio-temporal systems with heterogeneity in their structure and behavior have two major problems associated with them. The first one is that such complex real world systems extend over very large spatial and temporal domains and consume so many computational resources to simulate that they are infeasible to study with current computational platforms. The second one is that the data available for understanding such systems is limited because they are spread over space and time making it hard to obtain micro and macro measurements . This also makes it difficult to get the data for validation of their constituent processes while simultaneously considering their global behavior. For example, the valley fever fungus considered in this dissertation is spread over a large spatial grid in the arid Southwest and typically needs to be simulated over several decades of time to obtain useful information. It is also hard to get the temperature and moisture data (which are two critical factors on which the survival of the valley fever fungus depends) at every grid point of the spatial domain over the region of study. In order to address the first problem, we develop a method based on the discrete event system specification which exploits the heterogeneity in the activity of the spatio-temporal system and which has been shown to be effective in solving relatively simple partial differential equation systems. The benefit of addressing the first problem is that it now makes it feasible to address the second problem.

We address the second problem by making use of a multilevel methodology based on modeling and simulation and systems theory. This methodology helps us in the construction of models with different resolutions (base and lumped models). This allows us to refine an initially constructed lumped model with detailed physics-based process models and assess whether they improve on the original lumped models. For that assessment, we use the concept of experimental frame to delimit where the improvement is needed. This allows us to work with the available data, improve

the component models in their own experimental frame and then move them to the overall frame. In this dissertation, we develop a multilevel methodology and apply it to a valley fever model. Moreover, we study the model's behavior in a particular experimental frame of interest, namely the formation of new sporing sites.

Multilevel Methodology for Simulation Of Spatio-Temporal Systems With Heterogeneous Activity: Application To Spread of Valley Fever Fungus

Rajanikanth Jammalamadaka, Ph.D.
The University of Arizona, 2008

Director: Prof. Bernard P. Zeigler

Spatio-temporal systems with heterogeneity in their structure and behavior have two major problems associated with them. The first one is that such complex real world systems extend over very large spatial and temporal domains and consume so many computational resources to simulate that they are infeasible to study with current computational platforms. The second one is that the data available for understanding such systems is limited because they are spread over space and time making it hard to obtain micro and macro measurements . This also makes it difficult to get the data for validation of their constituent processes while simultaneously considering their global behavior. For example, the valley fever fungus considered in this dissertation is spread over a large spatial grid in the arid Southwest and typically needs to be simulated over several decades of time to obtain useful information. It is also hard to get the temperature and moisture data (which are two critical factors on which the survival of the valley fever fungus depends) at every grid point of the spatial domain over the region of study. In order to address the first problem, we develop a method based on the discrete event system specification which exploits the heterogeneity in the activity of the spatio-temporal system and which has been shown to be effective in solving relatively simple partial differential equation systems. The benefit of addressing the first problem is that it now makes it feasible to address the second problem.

We address the second problem by making use of a multilevel methodology based on modeling and simulation and systems theory. This methodology helps us in the

construction of models with different resolutions (base and lumped models). This allows us to refine an initially constructed lumped model with detailed physics-based process models and assess whether they improve on the original lumped models. For that assessment, we use the concept of experimental frame to delimit where the improvement is needed. This allows us to work with the available data, improve the component models in their own experimental frame and then move them to the overall frame. In this dissertation, we develop a multilevel methodology and apply it to a valley fever model. Moreover, we study the model's behavior in a particular experimental frame of interest, namely the formation of new sporing sites.

Chapter 1
Introduction

The study of methods to simulate systems which vary in both time and space is important as it can lead to the discovery of efficient methods which can exploit the spatio-temporal heterogeneity in the activity [11] of the system. One such methodology is based on the Discrete Event System Specification(DEVS) [29]. A method based on DEVS is inherently efficient in simulating systems which exhibit spatio-temporal heterogeneity in their activity behaviors. It does so by allocating more computational resources to the areas of high activity and less computational resources to the regions of low activity. On the other hand, a method which allocates its computational resources uniformly without regard to the spatio-temporal heterogeneity in the activity of a system is inefficient when compared with a method based on DEVS.

Valley fever is a disease which affects the lungs of an individual. It is prevalent in the Southwestern parts of the United States and Northern Mexico. It is caused by a fungus known as Coccidiodes. The fungus is hosted in soil and uses airborne spores as a means of propagation.

To start with, we had an agent-based implementation of the valley fever model [10] which used the NetLogo software [10]. As judged by domain experts (Dr. Gettings), the agent-based implementation was inaccurate and produced more than usual number of new sites in the simulation of the valley fever model. Another major limitation of this implementation was that it could run the valley fever model only for 23 years because of memory limitations and took about 10 hours to run [10]. To address this issue, we developed the DEVS counterpart for the valley fever model which runs much faster than the agent-based model while providing the same accuracy as the agent-based model. Since the DEVS counterpart runs much faster, we decided to develop more accurate process models in the DEVS framework. These physics-based process models were developed from first principles approach and with parameter tuning with the actual data. The DEVS model which includes the high resolution components now became a base model and it was studied for its behavior with respect to the experimental frame of formation of new sites in the valley fever model. But to study the base model in a 1500×1500 spatial region with variable favorableness was not feasible. So, we studied the behavior of the base and lumped models in a 200×200 spatial region which had binary favorableness for the growth of the valley fever fungus. We found out that the base model was more accurate and produced fewer new sites than the lumped model. But, the execution time of the base model was much higher than the execution time of the lumped model. So, we made an attempt to correct the parameters of the lumped model using the concept of parameter morphism.

1.1 Goals

The main goals of this dissertation were:

1. To apply the multilevel modeling methodology(concept of base and lumped models [28]) to the valley fever model and to experiment with parameter morphism to see if we can get the behavior of the lumped model as close as possible to the base model within a given experimental frame.

2. To show that a method based on DEVS is inherently efficient in simulating systems which exhibit spatio-temporal heterogeneity in their activity behaviors.

3. To show that the behavior of the agent-based implementation of the valley fever model can be reproduced by a counterpart DEVS model.

4. To develop physics-based moisture and temperature models for the valley fever model and study their effect on the behavior of the valley fever fungus with

respect to a given experimental frame.

5. To find an approximate relation between the diffusivity κ of the temperature model and the moisture content of the soil at a certain depth. In the absence of one of the parameters, this relationship can be used to get an approximate value for the missing parameter.

As will be shown in the later chapters, most of these goals have been met by:

1. Applying the concept of base and lumped models to the valley fever model and by showing that certain parameters can be varied to bring the behavior of the lumped model as close as possible to the base model within a specified experimental frame-the formation of new sites.

2. Showing that a method based on DEVS is efficient in simulating the valley fever model when compared with an equivalent agent-based implementation.

3. Showing that the behavior of the agent-based model can be faithfully reproduced by a counterpart DEVS model.

4. Developing physics-based moisture and temperature models for the valley fever model and by studying the effect of them on the behavior of the valley fever fungus with respect to the experimental frame of formation of new sites.

5. Finding an approximate relation between the diffusivity κ of the temperature model and the moisture content of the soil at a certain depth.

1.2 Previous Work

This section discusses prior work in the area of Modeling and Simulation and the previous agent-based implementation of the valley fever model.

1.2.1 Modeling and Simulation

A model which must be capable of accounting for all the input-output behavior of a real system and be valid in all allowable experimental frames can never be fully known. This model, called the base model, would be very complex and require such great computational resources that it would be almost impossible to simulate. However, given an experimental frame of current interest, a modeler is likely to find it possible to construct a relatively simple model that will be valid in that frame. This is a lumped model. It is the experimenter's image of the real system with components lumped together and interactions simplified. The concept of lumped and base models was applied to the predator-prey(Lotka-Volterra) model by Dr. Zeigler in [28].

1.2.2 Valley Fever Model

The previous implementation of the valley fever model [10] was based on an agent-based simulation and used the NetLogo [26] software. The ground model considered in this implementation had binary favorableness i.e. either the ground was favorable for the growth of the fungus or it was unfavorable for the growth of the fungus. The temperature model was simulated using a cosine function given in [4]. The wetting wave of the moisture model was simulated by using a rainfall model for the surface rain. The drying wave of the moisture model was simulated by computing the heat flow (the thermal conductivity of the soil multiplied by the temperature difference between soil-depths of 20 cm and 2.5 cm). Then, the amount of moisture corresponding to this heat flow was evaporated from the moisture present at the depth of 20 cm. As a result of the usage of coarse moisture, temperature and ground models, this implementation produced far more new sites(dormant + sporing sites) in the valley fever model. Another major limitation of this implementation was that it could only simulate the valley fever model for 23 years due to memory limitations and took 10 hours to run. This is because it did not exploit the spatio-temporal heterogeneity in the activity behavior of the valley fever model. There is spatial heterogeneity in activity as the ground has binary favorableness. There is temporal heterogeneity in activity as the factors on which the survival of the fungus depends(temperature, moisture, wind) are functions which vary with time. The behavior of the previous implementation of the valley fever model was similar to that of a discrete time model which allocates computational resources uniformly without regard to the heterogeneity in activity of the system.

1.3 Contributions of this dissertation

1.3.1 Modeling and Simulation

This dissertation extends the work done by Dr. Zeigler [28] and applies it to the valley fever model. Figure 1.1 shows the multi level methodology which is explained as follows. We have a spatio-temporal system to start with and it is abstracted into a base model which would consume a lot of computational resources and whose execution time is so long that it is not viable for practical usage. In order to address this issue, we specify an experimental frame of interest and convert the base model into a lumped model. In the process of this conversion, we preserve the behavior of the base model in the frame of interest and at the same time exploit the spatio-temporal heterogeneity in the activity of the spatio-temporal system.

Figure 1.2 shows the application of the multi level methodology to the valley fever model. Here the spatio-temporal system is the valley fever spread model. As mentioned before, the conversion of the Agent-based implementation of the valley fever model into the corresponding DEVS implementation results in a large reduction in the run time while preserving the behavior of the Agent-based implementation. Here the experimental frame is the formation of new sites(dormant+sporing sites) in the valley fever model.

In this dissertation, we apply the concept of base and lumped models [28, 29] to

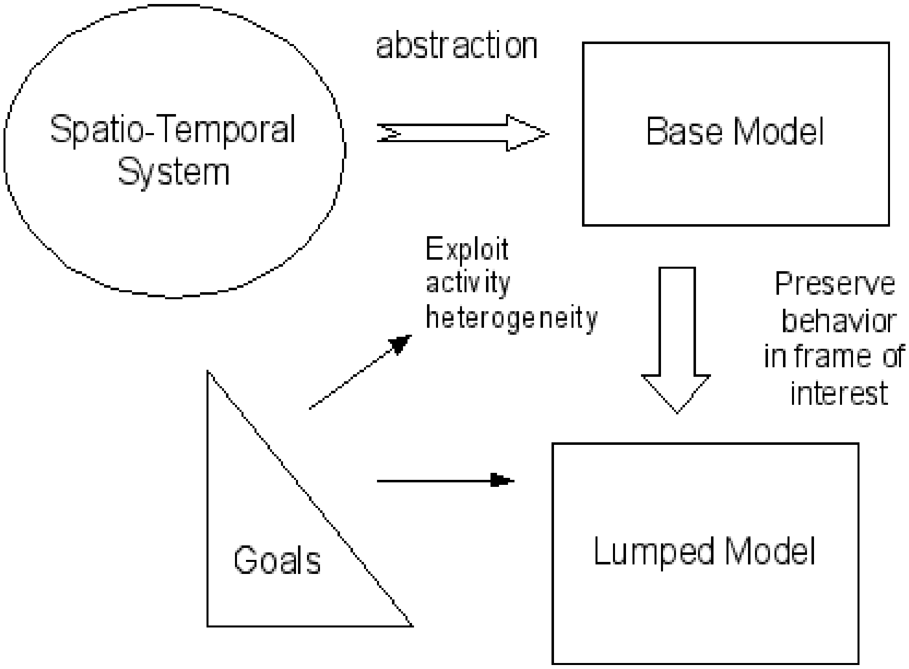

abstraction
Spatio-Temporal System
Base Model
Exploit activity heterogeneity
Preserve behavior in frame of interest
Goals
Lumped Model

FIGURE 1.1. The multi-level modeling methodology.

the valley fever model and we experiment with parameter morphism to see if we can get the behavior of the lumped model as close as possible to the base model within a given experimental frame-formation of new sites(dormant+sporing sites).

This dissertation also builds upon the work done by Zhang [30] and proposes a new activity-based partition algorithm and uses distributed simulation to apply this algorithm to the valley fever model.

1.3.2 Valley Fever Model

As mentioned before, the previous agent-based implementation of the valley fever model did not take advantage of the spatio-temporal heterogeneity that was present in the model. As a result of this, the agent-based implementation could simulate the

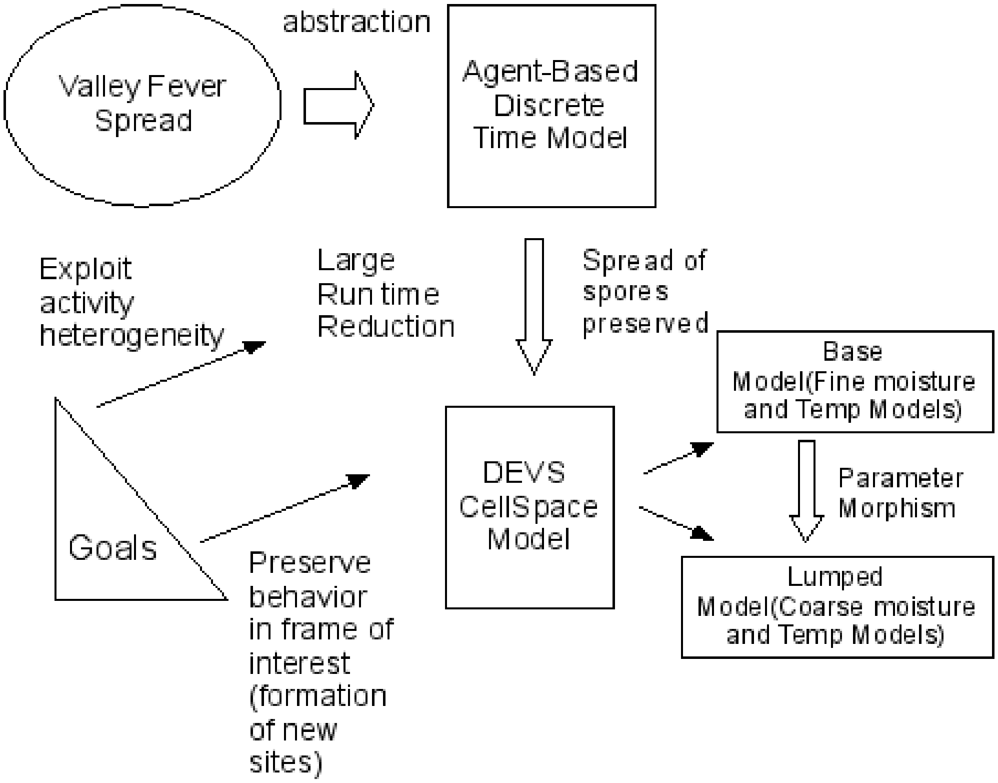

Valley Fever Spread
abstraction
Agent-Based Discrete Time Model
Exploit activity heterogeneity
Large Run time Reduction
Spread of spores preserved
Base Model(Fine moisture and Temp Models)
Goals
DEVS CellSpace Model
Parameter Morphism
Preserve behavior in frame of interest (formation of new sites)
Lumped Model(Coarse moisture and Temp Models)

FIGURE 1.2. An application of the multi-level modeling methodology to the valley fever model.

valley fever model only for 23 years due to memory limitations and took 10 hours to run. Several studies [11, 20, 21, 13, 12, 14] have shown us that a simulation methodology which is based on DEVS is inherently efficient in simulating models which exhibit spatio-temporal heterogeneity in their activity behaviors. This motivated us to reimplement the valley fever model using a simulation methodology that was based on DEVS. As will be shown, this resulted in a significant improvement in the execution time of the valley fever model.

The agent-based implementation of the valley fever model used a cosine function given in [4] in order to model the temperature. But, it was noticed that the rate of heating was higher than the rate of cooling. In order to accurately model this behavior, this dissertation contributes a new temperature model which is based on the works of [19] and [4, 5]. This new temperature model is validated by fitting it to a 2-year temperature data which was collected by Dr. Gettings [8]. The agent-based implementation used a ground model with binary favorableness. But, the actual ground has variable favorableness. This dissertation contributes a new ground model(by using MATLAB's fuzzy logic toolbox) which is based on the data collected by Dr. Bultman [9]. The agent-based implementation takes the rainfall data on the surface of the soil as input and computes the moisture at a depth of 20 cm by evaporating moisture that corresponds to the heat flow. But, this is a very crude approximation for computing the moisture. To address this issue, this dissertation contributes a new moisture model which is based on the works of [19, 4, 5] and on the data collected by Dr. Gettings [8]. As stated in the goals section, this was one of the major goals of this dissertation. According to domain experts such as Dr. Gettings, the usage of these new physics-based models should reduce the formation of new sites in the valley fever model. As will be shown, the use of the new models indeed reduces the formation of new sites in the valley fever model. Another contribution of this dissertation is to find an approximate relation between the diffusivity κ of the temperature model and the moisture content of the soil at a certain depth. This is useful because in most places it is easy to record the temperature but it is hard to record the moisture.

The main usage of the contributions of this dissertation are that they help the United States Geological Survey(USGS) in knowing the weather patterns(right combinations of the factors(temperature, moisture, wind) which affect the growth of the valley fever fungus) in the past which triggered an outbreak in the valley fever. This is very useful because it can help predict what kind of weather patterns to look forward to in the future which might result in an outbreak of the valley fever. In addition to that, the new moisture, temperature and ground models can be useful to anybody who wants to do epidemiology modeling of organisms which depend on the soil for their survival.

1.4 Organization of this dissertation

Chapter 2 gives a brief introduction to the Discrete Event System Specification (DEVS) and the concept of Activity [11].

Chapter 3 gives a detailed description of the valley fever model. It also explains in detail the old temperature, moisture and ground models and explains their limitations. Then, it explains the new temperature and moisture models and validates them by fitting them to the data collected by Dr. Gettings [8]. It also shows a brief comparison between a discrete time model and a model based on DEVS and shows the fundamental difference between them.

Chapter 4 explains the multilevel modeling methodology and shows its application to the valley fever model. It gives a detailed description of the concepts of lumped and base models and shows their application to the valley fever model within a specified experimental frame.

Chapter 5 discusses the compatibility of the base and lumped models within a specified experimental frame. It shows the performance advantage of the lumped model over the base model for the valley fever model by comparing their execution times.

Chapter 6 shows several results of running the lumped and base versions of the valley fever model by varying the parameters which affect the spread of the valley fever fungus. According to domain experts such as Dr. Gettings, the usage of the new moisture and temperature models should result in a significant reduction in the formation of new sites(dormant+sporing sites) in the valley fever model. This chapter shows results which validate the use of the new physics-based moisture and temperature models by verifying that their use indeed reduces the formation of new sites(dormant+sporing sites) in the valley fever model. It also explains the concept of parameter morphism and shows some experiments in which we try to bring the behavior of the lumped and base models as close as possible within a given experimental frame.

Chapter 7 extends the work done by Zhang [30] and proposes an activity-based partition algorithm for performing distributed simulation of the valley fever model. It then shows some results of applying this algorithm to the valley fever model and compares its performance with a method based on an algorithm which doesn't exploit the spatio-temporal heterogeneity in the activity present in the valley fever model.

Chapter 8 concludes this dissertation and discusses future work.

Chapter 2

Discrete Event System Specification-DEVS

This chapter gives a brief introduction to Discrete Event System Specification(DEVS). It explains the structure of DEVS and describes the basic constituent of any DEVS model- atomic model. Then, it describes a simplified simulation algorithm which is useful for simulating flat models. It concludes by briefly explaining the concept of activity and shows its significance in explaining the performance advantage of a DEVS based method over a method which allocates its resources uniformly without regard to the heterogeneity in the activity behavior of a system.

2.1 DEVS

This section provides a rudimentary introduction to DEVS. A thorough treatment can be found in [21, 29].

DEVS uses two types of structures to describe a discrete event system. Atomic models describe the behavior of elementary components. Coupled models describe collections of interacting components, where components can be atomic and coupled models.

An atomic model is described by a set of inputs, set of outputs, and set of states, a state transition function decomposed into three parts, an output function, and a time advance function. Formally, the structure is

$$M =< X, Y, S, \delta_{int}, \delta_{ext}, \delta_{con}, \lambda, ta >$$

where

X is a set of inputs,

Y is a set of outputs,

S is a set of states,

$\delta_{int} : S \rightarrow S$ is the internal state transition function,

$\delta_{ext} : Q \times X^b \rightarrow S$ is the external state transition function

with $Q = \{(s,e) \mid s \in S \,\&\, 0 \leq e \leq ta(s)\}$

and X^b is a bag of values appearing in X,

$\delta_{con} : S \times X^b \rightarrow S$ is the confluent state transition function,

$\lambda : S \rightarrow Y$ is the output function, and

$ta : S \rightarrow \mathbb{R}$ is the time advance function.

The external transition function describes how the system changes state in response to input. When input is applied to the system, it is said that an external event has occurred. The internal transition function describes the autonomous behavior of the system. When the system changes state autonomously, an internal event is said to have occurred. The confluent transition function determines the next state of the system when an internal event and external event coincide. The output function generates output values at times that coincide with internal events. The output values are determined by the state of the system just prior to the internal event. The time advance function determines the amount of time that must elapse before the next internal event will occur, assuming that no input arrives in the interim.

Coupled models are described by a set of components and a set of component couplings. Components of a coupled model can be atomic models and other coupled models. Just as with atomic models, coupled models have an interface defined in terms of input and output sets. The coupled model behavior is defined by its component models and their interconnections. DEVS is a modular modeling formalism, and so interactions with a coupled model must occur through its external interface; a coupled model is indistinguishable from an atomic model when viewed from the outside.

Coupled models are described by a set of components and a set of component output to input mappings. For this introduction, we restrict our attention to flat models (i.e., a structure composed entirely of atomic models) without external input or output coupling (i.e., the component models can not be affected by elements outside of the network). With these restrictions, a coupled model is described by the structure

$$N =< \{M_k\}, \{z_{ij}\} >$$

where

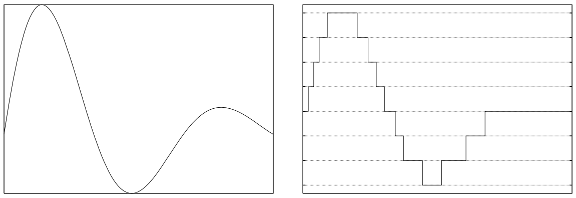
$\{M_k\}$ is a set of atomic models, and

$\{z_{ij}\}$ is a set of output to input maps $z_{ij} : Y_i \rightarrow X_j \cup \{\Phi\}$

where the i and j indices correspond to M_i and M_j in $\{M_k\}$ and Φ is the non-event.

The output to input maps describe how atomic models affect one another. The output to input map is, in this application, somewhat over generalized and could be replaced with a more conventional block diagram description. The non-event is used, in this instance, to represent components that are not connected. That is, if component i does not influence component j, then $z_{ij}(y_i) = \Phi$, where $y_i \in Y_i$.

These structures describe what a model can do. A canonical simulation algorithm is used to generate dynamic behavior from the description. Algorithm 1 is a simplified version of the complete simulation procedure. This algorithm is suitable for simulating flat models. Algorithm 1 assumes a coupled model N, with a component set $\{M_1, M_2, ..., M_n\}$, and a suitable set of output to input maps. For every component model M_i, there is a time of last event and time of next event variable, tL_i and tN_i respectively. There are also state, input, and output variables s_i, x_i, and y_i, in addition to the basic structural elements (i.e., state transition functions, output



(a) Continuous (b) Discrete state

Figure 2.1. Discrete state simulation of a continuous system.

function, and time advance function). The variables x_i is a bag, with elements taken from the input set X_i. The simulation time is kept in variable t.

The complete coupled model system specification and its simulation procedure can be found in [29]. Typically, DEVS modelers will rely on one of several DEVS simulation engines, many of which are available on-line. These simulation engines provide an object oriented interface for describing atomic and coupled models, and simulation classes for executing models that properly implement the interfaces.

A discrete state (or quantized) numerical integration scheme can be specified in terms of DEVS (see, e.g., [29], [21], and [15]). Discrete event integrators operate on a discretization of the system state space. Events correspond to jumps from one discrete state to the next. The time at which a jump occurs is approximated using the derivative of the system. Figure 2.1 demonstrates the end result of this discrete state approximation to a continuous process.

2.2 Activity

This section gives a brief overview of the concept of activity. For more details, the reader is referred to [11]. We say that something is active when there is a change in a particular parameter. The definition of the parameter may depend upon the context in which we want to define the activity. Activity as it relates to DEVS is defined as the rate of change of the parameter in the temporal and spatial dimensions. The

Algorithm 1 DEVS simulation algorithm.

```
$t \leftarrow 0$
for all $i \in [1, n]$ do
  $tL_i \leftarrow 0$
  set $s_i$ to the initial state of $M_i$
end for
while terminating condition not met do
  for all $i \in [1, n]$ do
    $tN_i \leftarrow tL_i + ta_i(s_i)$
    Empty the bag $x_i$
  end for
  $t \leftarrow min\{tN_i\}$
  for all $i \in [1, n]$ do
    if $tN_i = t$ then
      $y_i \leftarrow \lambda_i(s_i)$
      for all $j \in [1, n]$ & $j \neq i$ & $z_{ij}(y_i) \neq \Phi$ do
        Add $z_{ij}(y_i)$ to the bag $x_j$
      end for
    end if
  end for
  for all $i \in [1, n]$ do
    if $tN_i = t$ & $x_i$ is empty then
      $s_i \leftarrow \delta_{int,i}(s_i)$
      $tL_i \leftarrow t$
    else if $tN_i = t$ & $x_i$ is not empty then
      $s_i \leftarrow \delta_{con,i}(s_i, x_i)$
      $tL_i \leftarrow t$
    else if $tN_i \neq t$ & $x_i$ is not empty then
      $s_i \leftarrow \delta_{ext,i}(s_i, t - tL_i, x_i)$
      $tL_i \leftarrow t$
    end if
  end for
end while
```

activity of a time function is defined as the total change in the function over a closed time interval. The activity of a function $y(t)$ over an interval $[t_1, t_2]$ is given by

$$A(y(t), t_1, t_2) = \int_{t_1}^{t_2} \left| \frac{d}{dt} y(t) \right| dt \ . \tag{2.1}$$

The concept of activity can be generalized to the spatio-temporal domain. For example, the valley fever model has a spatial heterogeneity in its activity as the ground model has a variable degree of favorableness. Also, it has a temporal heterogeneity in its activity as the various factors which affect the growth of the fungus(moisture, temperature, wind) are functions of time. Several studies [11, 16, 17, 18] have shown that a simulation which is based on DEVS methodology can efficiently exploit such spatio-temporal heterogeneity in activities by focusing its computational resources on the regions of high activity.

Chapter 3

The Valley Fever Model

This chapter gives a brief description of the valley fever disease. Then, it gives a detailed description of the valley fever model. It also explains in detail the old temperature, moisture and ground models and explains their limitations. Then, it explains the new physics-based temperature and moisture models and validates them by fitting them to the data collected by Dr. Gettings [8]. It also shows an approximate relation between the diffusivity κ of the temperature model and the moisture content of the soil at a certain depth in the soil.

3.1 Valley Fever Disease

Valley fever is a disease which affects the lungs of an individual. It is prevalent in the Southwestern parts of the United States and Northern Mexico. It is caused by a fungus known as Coccidiodes. The fungus is hosted in soil and uses airborne spores as a means of propagation. The fungus form that can be airborne, the arthroconidia does not survive for long periods of time and relatively extreme conditions of temperature, humidity, and ultraviolet radiation, but under favorable conditions it can survive for a long period of time, especially in the soil [23, 10]. In this dissertation, we will refer to the arthroconidia as spores. In most cases, the fungus is asymptomatic. But, in the case of immunocompromised individuals, the treatment is very expensive and is a time consuming process. In some cases, it may be fatal.

3.2 Model Description

The model assumes that for a new site to become established, four factors must be simultaneously satisfied. First, there must be transport of spores from a source site to sites that are favorable for the growth of the fungus. Second, there must be sufficient moisture for fungal growth. Third, the temperature of the surface and soil must be favorable for growth. Fourth, the temperature and moisture must remain in favorable ranges for a long enough time interval for the fungus to grow down to depths at which spores at least will survive subsequent heat, aridity and ultraviolet radiation of the hot, dry season. The model starts with a given number of dormant sites. These will become active sources of spores when they receive enough moisture and have the correct temperature for the time required for the fungus to grow from a survival depth(20 cm in the model) in the soil up to the surface and produce spores. When this occurs for a dormant site it becomes a sporing site. A site becomes colonized when the wind blows from a sporing site to a favorable site, and there is at least a minimum amount of moisture in the ground at that site, and the temperature must be below the maximum for fungus growth. The colonized site will continue to grow until it runs out of moisture or the temperature becomes too warm for growth. If the site has grown long enough for the fungus to penetrate the soil to the survival depth, then the colonized site becomes a sporing site. A sporing site produces wind-borne spores. A colonized site dies if it does not have enough moisture or gets too hot before the growth interval to become a permanent site passes. A sporing site becomes dormant when there is too little moisture or the temperature gets too high, killing the fungus down to the survival depth in the soil. Figure 3.1 shows the state transition diagram for the valley fever fungus.

3.3 Ground Model

The favorableness of the ground was determined by using a fuzzy logic model [9] taking into account the following factors : [the] soil type, slope, aspect, presence of bedrock and clay content. MATLAB's fuzzy logic toolbox was used for this purpose. The spatial heterogeneity in the favorableness can be easily noticed from figure 3.2.

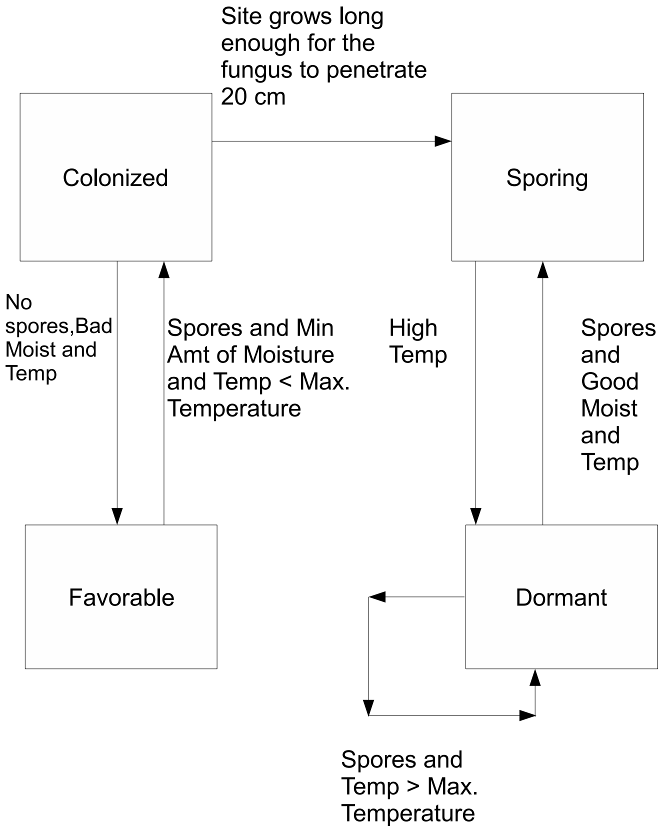

Site grows long enough for the fungus to penetrate 20 cm
Colonized
Sporing
No spores,Bad Moist and Temp
Spores and Min Amt of Moisture and Temp < Max. Temperature
High Temp
Spores and Good Moist and Temp
Favorable
Dormant
Spores and Temp > Max. Temperature

FIGURE 3.1. State transition diagram for the valley fever fungus

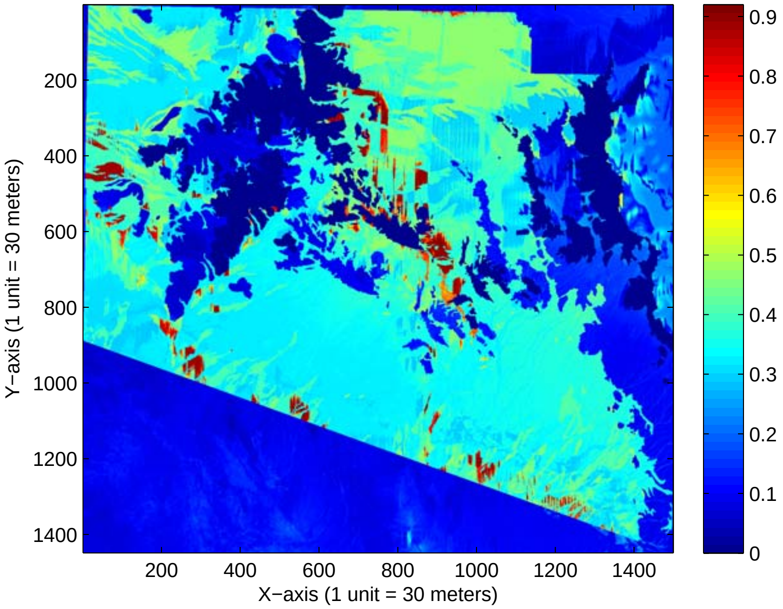


FIGURE 3.2. Favorableness of the ground.

Figure 3.2 shows the resulting ground model for the Organ Pipe National Monument in Southwestern Arizona with the scale ($[0, 1]$) on the right denoting the degree of favorableness of the ground for the growth of the valley fever fungus. In the scale, the number 0 represents least favorableness and the number 1 represents highest favorableness for the growth of the fungus.

Tables 3.1, 3.2, 3.3 show some of the rules that were used in determining the degree of favorableness for the ground model.

TABLE 3.1. Slope to favorableness conversion table.

Slope	Favorableness
Very Low $(0 - 3)$	Very High
Low $(4 - 7)$	High
Medium $(8 - 15)$	Medium
High $(15 - 25)$	Very Low
Very High (> 25)	Very Low

Table 3.2. Aspect to favorableness conversion table.

Aspect	Favorableness
East (45 – 135)	High
South (135 – 225)	Very High
West (225 – 315)	High
North (315 – 45)	Medium

Table 3.3. Clay Content to favorableness conversion table.

Clay Content	Favorableness
Very Low	Very Low
Low	Low
Medium	Very High
High	Low
Very High	Very Low

The favorable region is divided into sites or patches for the purposes of computation. Each patch can be in any of the following states: favorable, colonized, sporing or dormant.

3.4 Temperature Model

Figure 3.3 shows a cosine fit [10, 4] and an average daily temperature for day of the year, 1894-2001, for Tucson, Arizona. The coarse version of the valley fever model used this cosine fit to model the temperature.

The data in figures 3.4, 3.5 and 3.6 was obtained from the foothills(Latitude: $32\,^{\circ}21'5.0''$, Longitude: $110\,^{\circ}57'44.8''$ and Elevation: 2758.4 feet) region of Tucson, Arizona [8]. It was collected using temperature sensors that were buried at depths of 2.5 cm, 20 cm and 29.2 cm respectively [8]. It can be seen from figures 3.4, 3.5 and 3.6 that the actual rate of cooling is less than the rate of heating. In order to model this

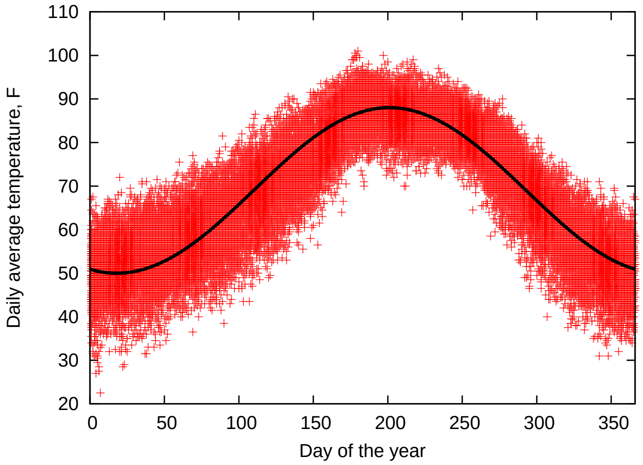


FIGURE 3.3. A cosine fit for the average daily temperature for day of the year, 1894-2001, for Tucson, Arizona.

behavior, the accurate temperature model was built using the error function solution given in [4]. Equation 3.1 gives the temperature at depth x and time t.

$$v(x,t) = \frac{x}{2\sqrt{\pi\kappa}} \int_0^t \phi(\lambda) \frac{e^{\frac{-x^2}{4\kappa(t-\lambda)}}}{(t-\lambda)^{3/2}} d\lambda \tag{3.1}$$

where $\phi(t)$ is the surface temperature, κ is the diffusivity and $v(x,t)$ is the temperature at a depth x and time t.

Equation 3.1 can be approximated as a sum of step function solutions [25] i.e. the transient temperature field $T(z_j,t)$ at an arbitrary depth z_j may be expressed in terms of a series of step changes ΔT_i over the time interval t_p (present) to t_i as shown in equation 3.2

$$T(z_j,t_p) = \bar{T}0 + \sum_{i=0}^{t_p-1} \Delta T_i erfc\left(\frac{z_j}{\sqrt{4\kappa(t_p-t_i)}}\right) \tag{3.2}$$

In equation 3.2, $\bar{T}0$ denotes the record of surface temperatures averaged over the data collection interval and is the approximation of the $\phi(\lambda)$ function in equation 3.1. $\bar{T}0$ is the input to the accurate temperature model. The temperature data from the surface and at depth in the soil is used to determine the thermal diffusivities

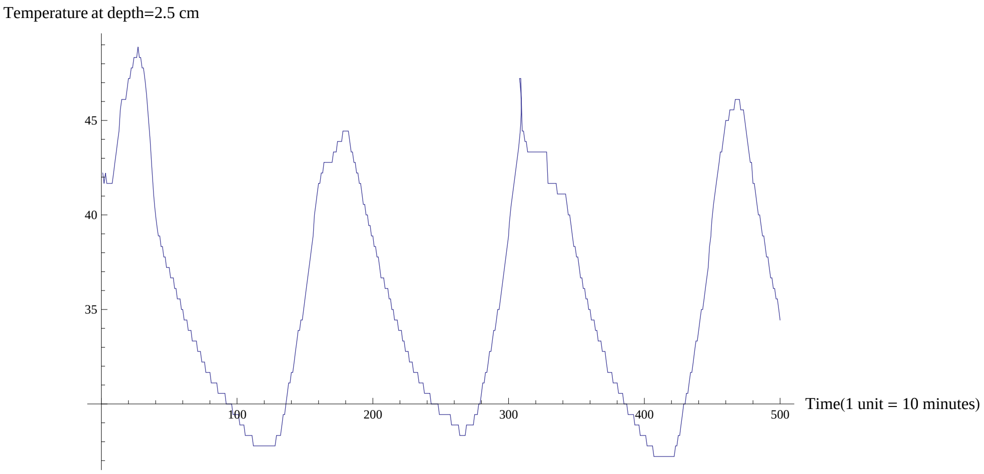


Figure 3.4. Temperature at depth = 2.5 cm at a site near the foothills region of Tucson, Arizona.

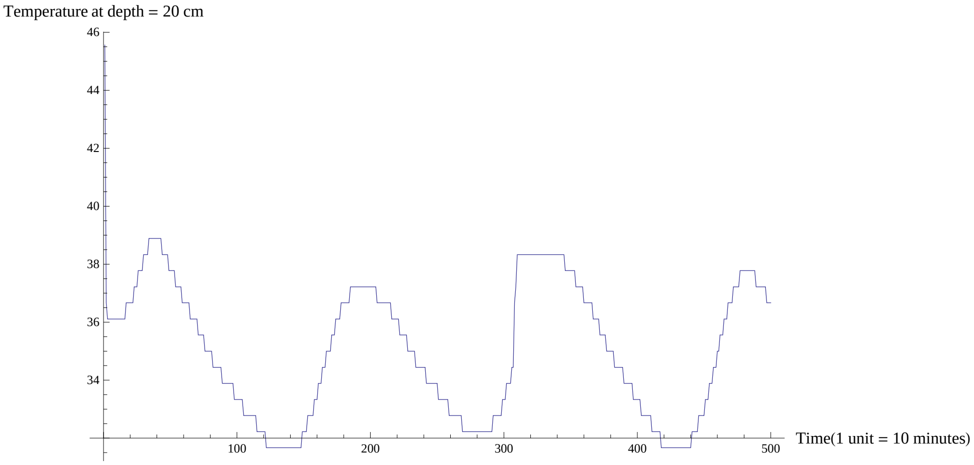


Figure 3.5. Temperature at depth = 20 cm at a site near the foothills region of Tucson, Arizona.

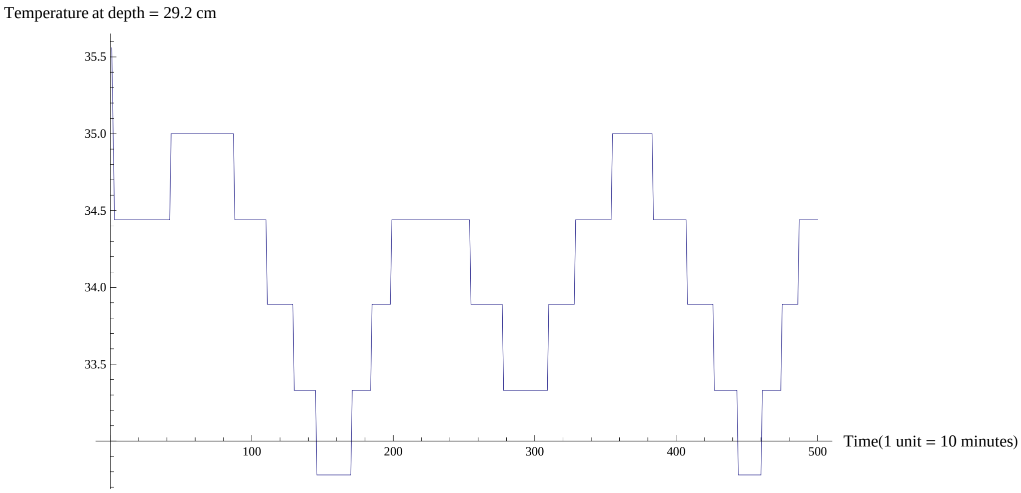


FIGURE 3.6. Temperature at depth = 29.2 cm at a site near the foothills region of Tucson, Arizona.

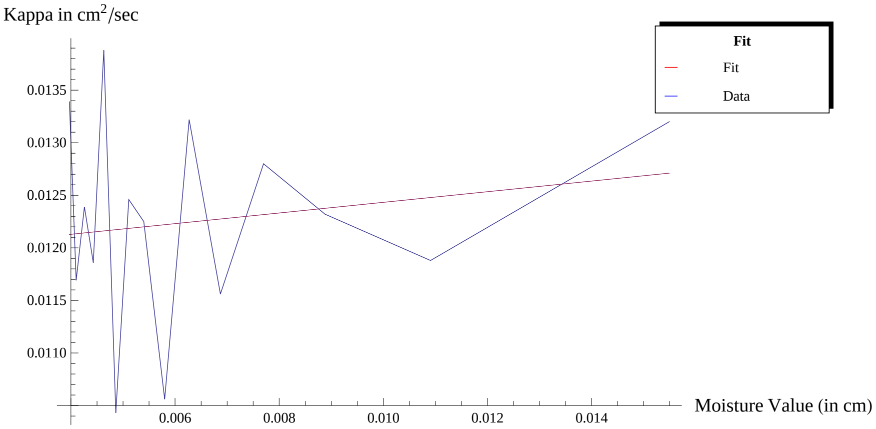


FIGURE 3.7. Fit for Kappa as a function of moisture content.

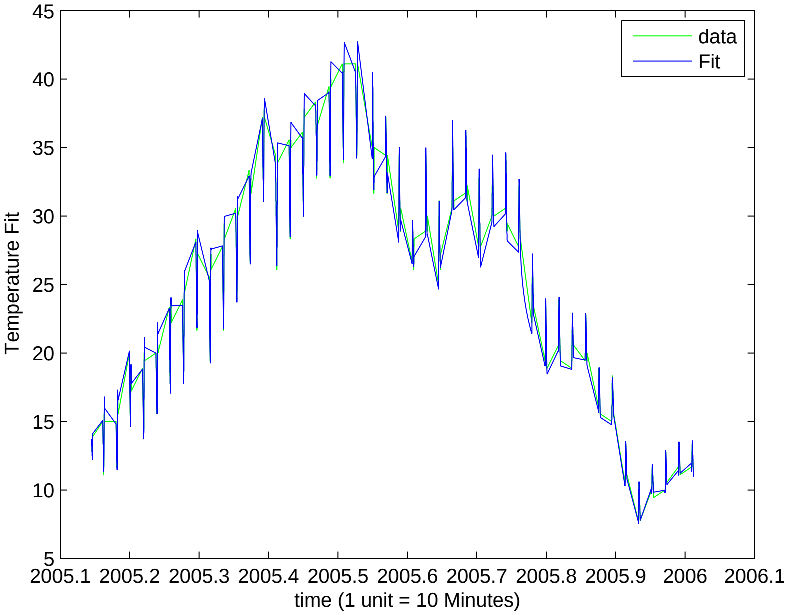


Figure 3.8. Fit for temperature at a depth of 20 cm when the temperature at a depth of 2.5 cm was given as input to the accurate temperature model.

Table 3.4. Kappa and Moisture Values.

Moisture Content (in cm) at a depth of 20 cm	κ $(\frac{cm^2}{sec})$
0.0154	0.0132
0.0109	0.01188
0.0088	0.01232
0.0077	0.0128
0.0068	0.01156
0.0062	0.01322
0.0058	0.01056
0.0054	0.01225
0.0051	0.01246
0.0048	0.01043
0.0046	0.01388
0.0044	0.01186
0.0042	0.01239
0.0041	0.01169
0.0039	0.01339

used in the model. As figure 3.7 shows, the variations in the thermal diffusivity are approximately explained by moisture content variations. Table 3.4 shows the κ and moisture values that have been used to come up with the fit function given by equation 3.3.

$$\kappa(moisture) = 0.05 * moisture + 0.012 \tag{3.3}$$

The $\chi - square$ fit for the observed temperatures was obtained by using the Fortran code given in [3]. Figure 3.8 shows the fit for the temperature at a depth of 20 cm when the temperature at a depth of 2.5 cm was given as input ($\bar{T}0$) to the accurate temperature model. The mean $\chi - square$ value for the fit is 0.46.

3.5 Wind and Precipitation Model

Wind plays an important role in the valley fever model as it is the main carrier of the spores and hence it decides what part of the ground model will be most active [11]. Wind and precipitation are modeled using random variables. Wind can have a preferred azimuth and a wedge angle which determines the width in which the wind can blow. Precipitation in the model occurs as a rainstorm with an azimuth and a specified width. Rainstorm azimuths are isotropic in this model. Wind occurrence, direction, and intensity are based on observed probabilities for the Tucson, Arizona area, as are precipitation occurrence and amplitude for the period 1894-2001.

3.6 Moisture Model

We modeled the moisture content of a patch by keeping track of the amount of rain on the surface for the patch and by using equation 3.5 which gives the moisture content at any depth (see [5, 4, 19]). This model was derived by starting with the diffusion equation $u_t = D * u_{xx}$(where u is the state variable and D is the diffusivity, [4]) with the boundary condition $u(0,t) = constant$ and initial condition given by $u(x,0) = 0$ $\forall x$ [19].

Table 3.5. Tensiometer to Volumetric Water Content conversion table.

Tensiometer Reading (centibars)	Volumetric Water Content (Inch/Inch)
0	1
2	0.30
5	0.26
9	0.21
17	0.17
26	0.15
37	0.12
51	0.12
200	0.0

Table 3.5 was used to convert the Tensiometer readings to the volumetric water content [7]. This table was used to find an empirical conversion formula which will convert a given tensiometer reading to the corresponding volumetric water content. The formula is given by equation 3.4.

$$\begin{aligned} function(tensiometer) &= 0.153241 + 0.187131 \times e^{-0.1129 \times tensiometer} \\ &- 0.0007 * tensiometer \end{aligned} \tag{3.4}$$

The data used to find this model was collected from the foothills (Latitude: 32°21′5.0″, Longitude: 110°57′44.8″ and Elevation: 2758.4 feet) region of Tucson, Arizona [8]. It was collected by putting Irrometers (moisture sensors) at depths of 20 cm and 29.2 cm [8]. We modeled the amount of the moisture present at a certain depth in the soil using two diffusive waves from the surface of the soil. This model was derived by approximating the moisture content as a function of depth and time by the sum of two diffusion waves - a wetting wave from recent rain events propagating downward, followed by a drying wave propagating downward at a later time.

This model is explained as follows. Let t_1 be the time when the rain starts falling on the surface of the soil, let t_2 be the time when the rain stops falling on the surface of the soil and lets assume that t_2 is also the time when it starts evaporating from the certain depth in the soil. Let Di denote the diffusivity of the rain water into the soil and De denote the diffusivity of the drying wave. Let τ denote the amount of time(in days) to look back from the current time t in order to account for the past rain events. Then, the moisture at a depth x in the soil as a function of depth and time is given by equation 3.5.

$$\begin{aligned} moisture(x,t) &= \sum_{i=t-\tau}^{t}\sum_{j=t1}^{t2}\text{rainfall}(j)\Big(\ \text{erfc}\Big(\frac{x}{2*\sqrt{600*Di*(t-t_{1i})}}\Big) \\ &- \text{erfc}\Big(\frac{x}{2*\sqrt{600*Di*(t-t_{2i})}}\Big) \\ &+ \text{erf}\Big(\frac{x}{2*\sqrt{600*De*(t-t_{2i})}}\Big)\Big) \end{aligned} \tag{3.5}$$

In equation 3.5, rainfall denotes the amount of rain that fell on the surface of the soil, t_{1i} and t_{2i} denote the start and stop times of the rain for the i^{th} rain event in the interval $[t-\tau,t]$. These start and stop times were determined by splitting the surface rainfall data. Any two rain events with a time difference of a couple of hours were treated as different. The reason for the factor of 600 is explained as follows. The data which we got from the site was separated by 10 minute time intervals. Since we wanted Di and De in units of $\frac{cm^2}{sec}$, we multiplied the time differences by a factor of 600 (10 minutes = 600 seconds). A depth of 20 cm and a τ value of 11 days were used to validate the moisture model. The model was fitted to the actual data using the NMinimize function of Mathematica and the mean Di and De were found to be $Di = 0.451\frac{cm^2}{sec}$ and $De = 0.183\frac{cm^2}{sec}$. The data and the fit are shown in figure 3.9. The fit that we got matches the data at most of the places but at some places it dries out completely when the data does not.

The moisture content of the patch is thus increased by rain on the patch and

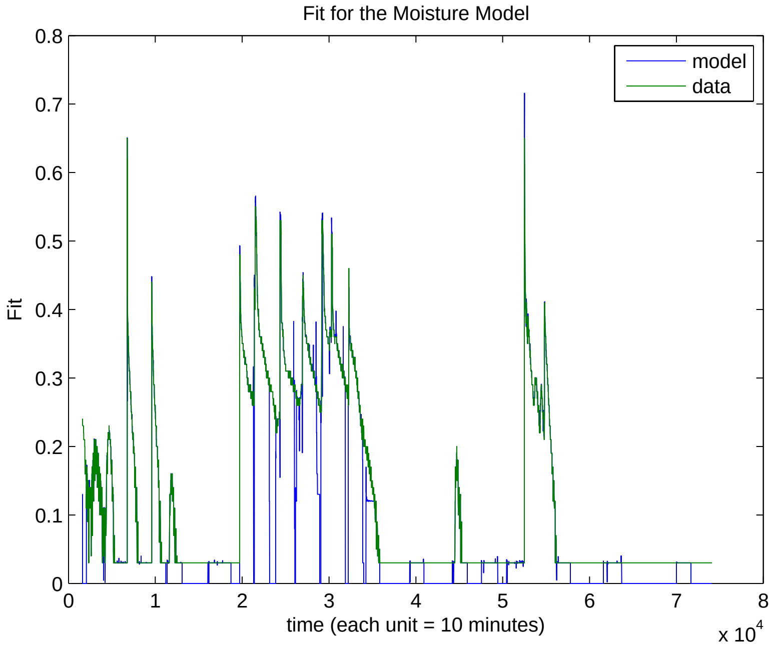


FIGURE 3.9. Fit for the moisture model.

decreased by heat flow, which causes evaporation. The moisture was modeled by heat flow with forced convection before, but now we model it based on the fit to data as shown by figure 3.9.

3.7 Description of the Discrete Time Model

In the discrete time valley fever model [26], the state of a site is computed at each time step. The new state of the site is a function of the current states of the site and its neighbors and of the inputs that the site receives from the wind and rain models. The time step is in hours; the simulation interval is in years. If the number of total sites (both favorable and unfavorable) is N, the total time of simulation T and the time step is Δt, then the total number of transitions made by the discrete time model is equal to

$$N * \frac{T}{\Delta t}$$

For a typical run

$$N = 10000 \text{ sites}$$

$$T = 5 \text{ years}$$

$$\Delta t = 1 \text{ hour}$$

the number of transitions made by the discrete time model is $4.38 * 10^8$.

3.8 Description of the DEVS Model

In contrast to the discrete time model, processing in a discrete event model takes place in the state space. We observed that in the discrete time model most of the sites are unfavorable and hence no state changes take place at those sites. Discrete event formulation of such a model is naturally more efficient than the corresponding discrete time version because it schedules events only at the sites that are favorable for the growth of the fungus. For example, in a discrete event implementation of the valley fever model, if spores fall on a site that is favorable, the model knows when to schedule the next event since it knows the time required for a site to become a sporing site. If the conditions are unfavorable the model does not need to schedule any further events. This results in a considerable reduction in computational effort in simulating models which exhibit spatio-temporal heterogeneity in their behaviors.

Chapter 4

Multilevel Modeling- Description of the Base and Lumped Model

This chapter explains the multilevel modeling methodology and shows its application to the valley fever model. It gives a detailed description of the concepts and lumped and base models and shows their application to the valley fever model within a specified experimental frame- spread of the valley fever fungus.

A model which must be capable of accounting for all the input-output behavior of a real system and be valid in all allowable experimental frames can never be fully known [27].This model, called the base model, would be very complex and require such great computational resources that it would be almost impossible to simulate. For the valley fever model, the base model can never be fully known because of the complexity of the system and the impossibility of observing all possible states. However, given an experimental frame of current interest(for example the number of sporing sites after a certain time period), a modeler is likely to find it possible to construct a relatively simple model that will be valid in that frame. This is a lumped model. It is the experimenter's image of the real system with components lumped together and interactions simplified [27, 28].

Modeling an ecosystem requires knowledge of the real system, obtained with experiments, and its abstraction within a mathematical framework. Systems methods can be used effectively in the latter phase of model development. Indeed, when coupled with experimental work, system-theoretic concepts can help in the development of an ecologically realistic mathematical model. The *state space approach* is the most widely used in modern systems analysis because it allows description of both observable and unobservable variables. The models are memoryless and nonanticipatory and the state of the system is predicted using information on the present state and inputs to the system [28].

The organization of the simulation models is accomplished by considering several elements and their relationships. These elements are a collection of experimental frames, the real system, and the domain of possible models. The experimental frames specify restrictions on experimental access to the real system. The models are assumed to be transition systems which are specifiable at various levels of structure and behavior within short-hand conventions(e.g., discrete event, and differential equation formalisms). Thus, the models are the various distributed and lumped models which can be postulated to account for the observed data and to predict the results of future experiments [28].

Ecosystems, as examples of large-scale multifaceted systems, require that a multiplicity of models be developed since a single all-encompassing model, however desirable as a conceptual goal, is not a practical object. By decomposing questions and modeling objectives into an ordered structure of elements called experimental frames [27], useful partial models may be constructed, validated, and employed, each one attuned to a particular experimental frame. Concomitant with the pluralism of such partial models is the recognition that models are expressible in different formalisms, each offering conceptual and computational advantages within its domain of application [28].

Figure 4.1 shows the base and lumped models for the valley fever model. The version of the valley fever model consisting of the fine resolution temperature, ground and moisture models constitutes the base model. The version of the valley fever model consisting of the low resolution temperature,ground and moisture models constitutes the lumped model.

Figure 1.2 shows the detailed view of the application of the multilevel modeling methodology to the valley fever model. The base model constitutes of the fine moisture and temperature models of the valley fever model. The lumped model con-

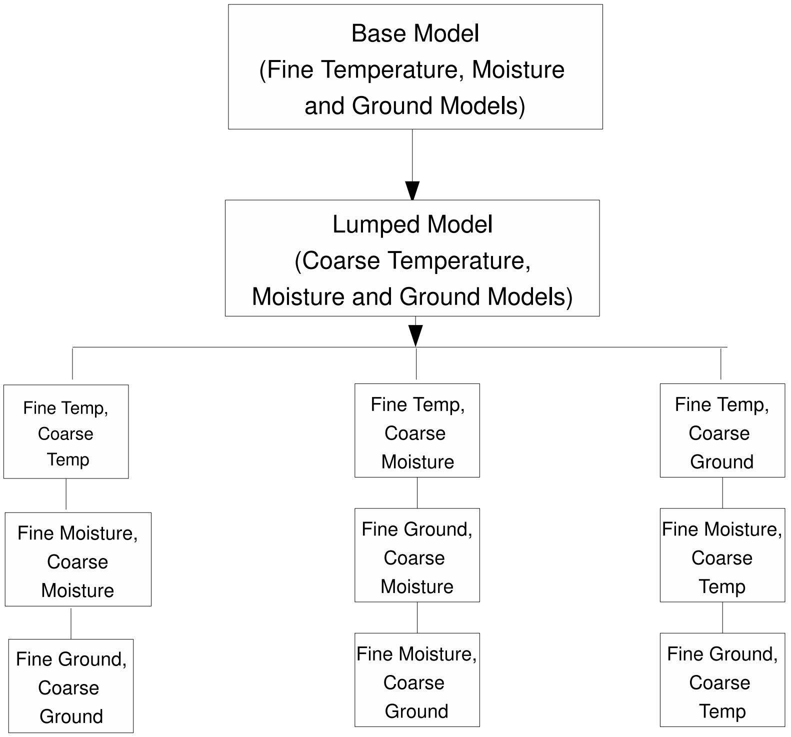


FIGURE 4.1. Base and lumped Models for the valley fever model.

stitutes of the coarse moisture and temperature models. In both the lumped and base versions of the valley fever model we exploit the spatio-temporal heterogeneity in its activity behavior. While doing the conversion of the base model to the lumped model using the concept of parameter morphism we try to bring the behaviors of the base and lumped models as close as possible to each other withing a specified experimental frame. In the case of the valley fever model, the experimental frame is the formation of new sites(dormant+sporing sites).

The lumped model is of low resolution as it doesn't deal with the nitty-gritty details of the composite model. It provides a high level overview of the whole model and is useful when a user wants to know the general behavior of a system in a short amount of time. In other words, a lumped model provides a qualitative picture of the behavior of a model. It is also computationally less taxing as all the models of the composite model are in a coarse form and as a result do not need lot of computations to be performed. The main disadvantage of using a lumped model is that the final result is of low fidelity and therefore is not useful when the user needs very accurate results.

The base model is of high resolution as it deals with the nitty-gritty details of the composite model. It provides a low level view of the whole model and is useful when a user wants to know the exact behavior of a system. The main advantage of the base model over the lumped model is that it provides results of high accuracy. In other words, in addition to providing a qualitative picture, a base model also provides a quantitative picture of the behavior of a model. On the flip side, it is computationally very taxing as all the models of the composite model are in a fine form and as a result they do need a lot of computations to be performed. The final result is of high fidelity and therefore is useful when the user needs very accurate results.

For example, consider the lumped and base model versions of the valley fever model. As will be shown in the results section, if the user is only concerned about the overall(qualitative) spread/survival of the valley fever fungus, then he is much better off using the lumped version of the valley fever model as it is less demanding on the computational resources. It also runs much faster than the base version of the valley fever model. On the other hand, if the user is interested in knowing the sites at which the fungus survived or what phase the fungus is in at that particular site, he needs to go for the base model. The only disadvantage of the base model is that it is computationally very demanding and time consuming.

4.1 Organization

The following are the concepts that are needed for the organization and integration of the above models [28]:

1. ξ - a collection of *experimental frames*. A frame $E \in \xi$ represents a restricted set of questions by specifying the restrictions on experimental access to the real system sufficient to answer them. Such a frame E determines a collection of data sets $\Delta(E)$, such that $D \in \Delta(E)$ is an *apriori* possible result of complete data acquisition within frame E.

2. R- the *real system*, comprises the specific data that have been, or would be, collected by experimenting with the system.

3. μ - *the domain of possible models*. These are assumed to be transition systems which are specifiable at various levels of structure and behavior and within various short-hand conventions such as the discrete event, and differential equation formalisms.

Roughly, the *experimental frames* will encode the various choices of observables (for example number of new sites in the valley fever model) and conditions (initial number of dormant sites, direction of Wind) under which experiments were run. The *real system* is the data collectible by making the implied observations under the given conditions. Finally, the *models* are the various distributed and lumped models which can be postulated to account for the observed data and to predict the results of future experiments.

In an ideal situation for each frame $E \in \xi$ there is a known model $M \in \xi$ which best answers the questions posable in E. By best we mean that the model can reproduce without error the data set $\Delta(E)$ in a manner which requires the least consumption of computer resources. Realistically, this ideal is not realizable after a necessarily finite span of data acquisition. The dynamics of modeling concern successive approximations to the ideal.

4.2 Experimental Frames

There are four main types of experimental frames. These types are distinguished by the descriptors: global, local, total and occupancy. The global descriptor refers to the fact that all cells(all the favorable locations in the valley fever model where the fungus may grow) in the model are being observed. On the other hand, in the local condition, only some subset of the cells are of interest. The total descriptor refers to the fact that the quantities of interest in a frame have been totaled to produce aggregate quantities, so that only these aggregates are observable in the frame. Finally, the occupancy descriptor refers to the fact that a frame permits only the observation of discrete occupancy states, such as whether or not a cell is favorable, whether or not a new site had been established on the cell, and whether or not that cell is producing any spores.

The occupancy frames are the most restricted. But still they permit consideration of the persistence of new sites since to determine whether or not there are any new sites requires only a binary consideration(whether that site is favorable/unfavorable for the growth of the fungus). On the other hand, the global frames permit observation of detailed spatial distribution of sites. The total frames correspond to classical populations in which spatial structure has been averaged out. It is obvious that certain frames are more informative than others.

Each experimental frame names a set of variables of interest, called the *compare* variables, and a set of variables determining the conditions under which experiments are to be performed, called the control variables. The most inclusive frame, $E^{global}_{sporing,colonized,dormant}$ specifies as compare variables: number of sporing, colonized and dormant cells. There are no control variables for this frame. An example of a frame which has a nonempty set of control variables is $E^{global}_{\frac{sporing,colonized,dormant}{wind,temperature,moisture}}$. Here, the set (sporing,colonized,dormant) are the phases in which a given cell can be and the set (wind,temperature,moisture) are the control variables which determine in what phase a given cell can be in.

In a frame, the compare variables readings are recorded against time for the duration of any particular experiment. This yields a time function also called a segment, or a trajectory which is referred to as a *data element* belonging to the frame [28]. Figures 4.2, 4.3, 4.4, 4.5, 4.6 show the number of dormant, sporing, dormant plus sporing, colonized and favorable sites as a function of time for a 25 year simulation run of the valley fever model whose ground model has binary favorableness.

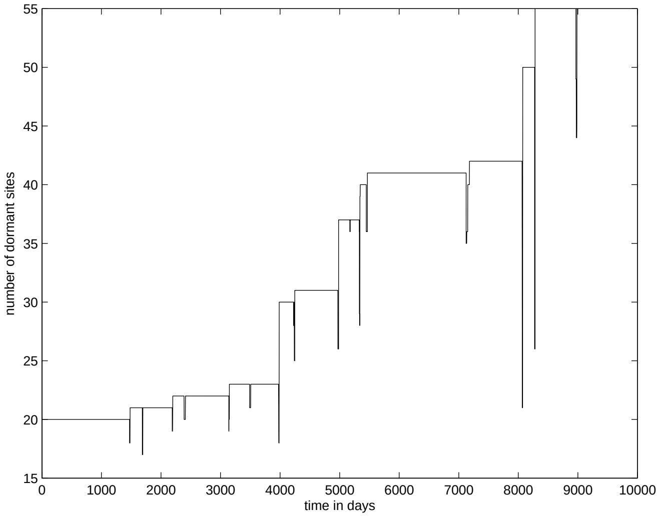


FIGURE 4.2. Number of dormant sites as a function of time.

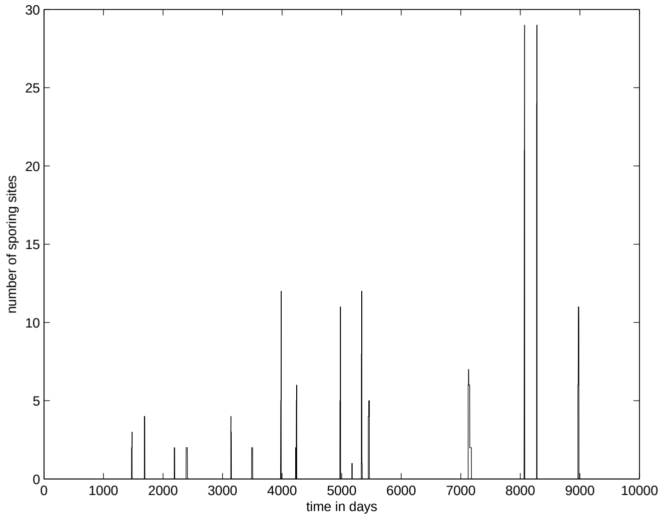


Figure 4.3. Number of sporing sites as a function of time.

The next chapter explores the possible combinations of the constituent models of the valley fever model and discusses their compatibility in relevance to the spread of the valley fever fungus.

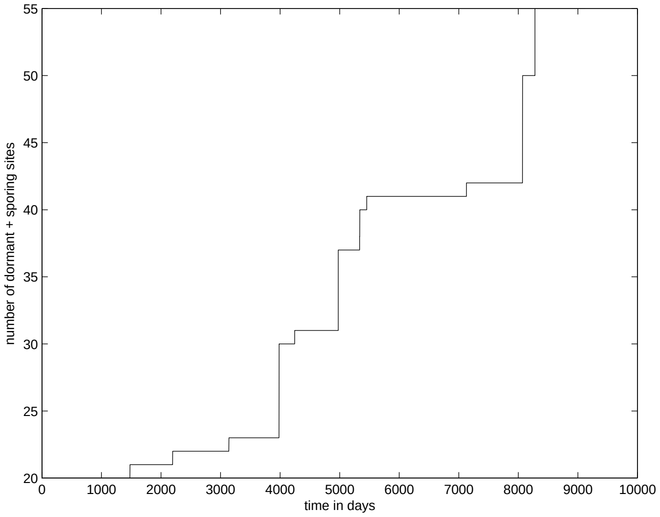


FIGURE 4.4. Number of dormant and sporing sites as a function of time.

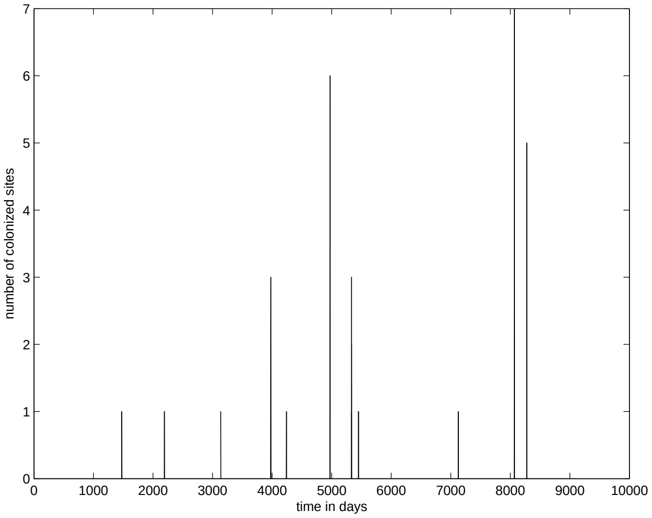


FIGURE 4.5. Number of colonized sites as a function of time.

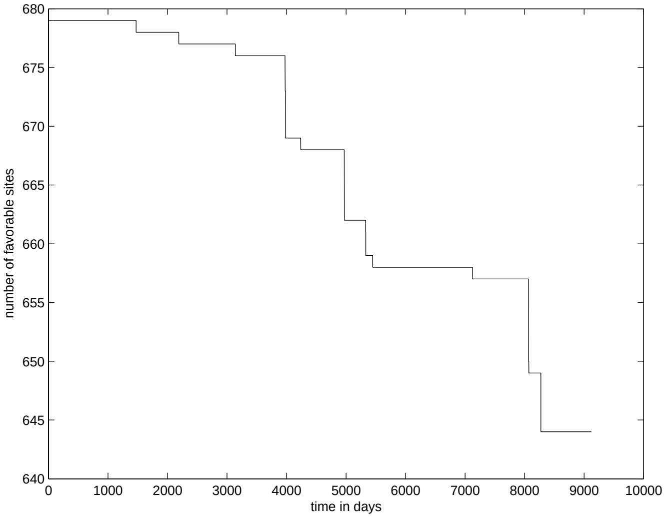


Figure 4.6. Number of favorable sites as a function of time.

Chapter 5
Comparison of Base and Lumped Models with respect to Experimental Frame

This chapter begins by comparing the agent-based implementation of the valley fever model with its counterpart DEVS model. As mentioned before, the agent-based implementation uses coarse moisture, temperature and ground models. According to domain experts such as Dr. Gettings this results in the formation of far more new sites(dormant+sporing sites) in the simulation of the valley fever model. In order to produce the counterpart DEVS model(lumped model), we used the same coarse moisture, temperature and ground models that have been used by the agent-based model. As a result, we see that the use of lumped model also produces far more new sites(dormant+sporing sites) in the simulation of the valley fever model. Then, this chapter describes the compatibility of the base/fine and lumped/coarse models within a specified experimental frame. It shows the performance advantage of the lumped model over the base model for the valley fever model by comparing their execution times.

5.1 Compare agent-based implementation of the valley fever model with its counterpart DEVS model

In order to validate the DEVS model with the agent based implementation we considered two test cases. Figures 5.1 and 5.2 show the results of simulating the agent-based and DEVS models using the test cases in columns 1 and 2 of table 5.1 for 20 years using a 200×200 ground model with binary favorableness. In the agent based model which uses the NetLogo software [26, 10], a brown color denotes a dormant site, a black color denotes a site which is unfavorable for the growth of the valley fever fungus, a yellow color denotes a site which is favorable for the growth of the valley fever fungus and a red color denotes a site which is in the sporing phase. The swaths represent rain storms on the surface of the soil.

Table 5.1. Parameters for comparing agent-based model with its counterpart DEVS model.

Parameter	1	2
wjmax	5	1
tradeWindAz	0	271
windAZRange	60	89
tempMax	40	30
moistMin(inches)	0.2	0.4
SiteMin(days)	9	9

In the DEVS model, a purple color denotes a dormant site, a white color denotes a site which is unfavorable for the growth of the valley fever fungus, a yellow color denotes a site which is favorable for the growth of the fungus and a red color denotes a site which is in the sporing phase.

In the first test case, the wind is blowing towards North. As a result, we see that in both the implementations(agent-based and counterpart DEVS implementation), the spread of the valley fever fungus is prominent in the Northern direction.

In the second test case, the wind is blowing towards East. As a result, we see that in both the implementations(agent-based and counterpart DEVS implementation), the spread of the valley fever fungus is prominent in the Eastern direction.

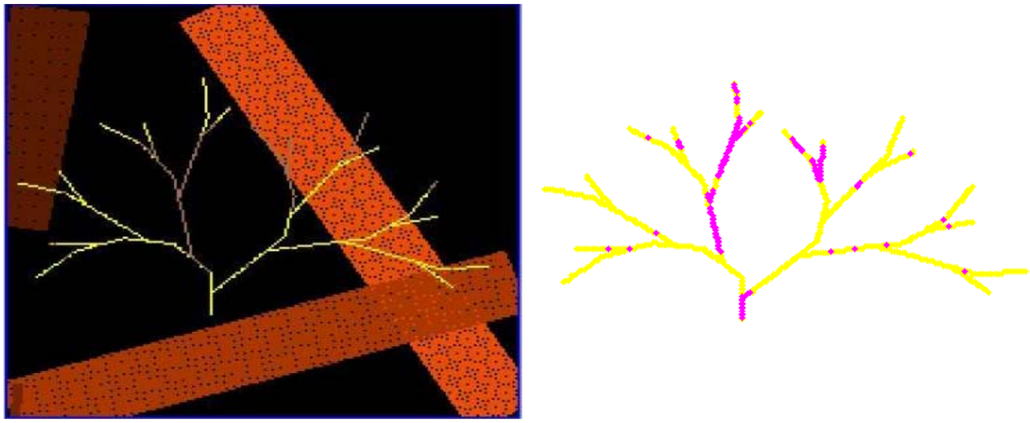
Table 5.2 shows the total number of new sites(dormant+sporing sites) formed by using the agent-based and DEVS implementation for the valley fever fungus.

As can be seen from table 5.2, the output of the DEVS model is comparable to that of the agent-based model.

Table 5.3 shows the execution times of the agent-based model [10] and the counterpart DEVS model as the simulation time is varied. From table 5.3, it can be seen

Table 5.2. New sites(dormant+sporing sites) for the agent-based model and its counterpart DEVS model.

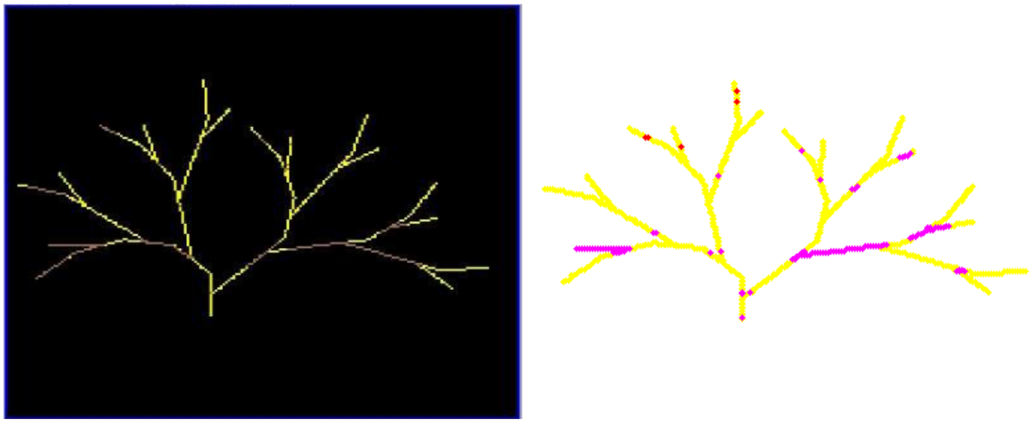
Test Case	Agent-Based Implementation	DEVS Implementation
1	167	187
2	207	185



(a) Agent Based Model

(b) DEVS Model

Figure 5.1. Outputs of Agent Based and DEVS models.



(a) Agent Based Model

(b) DEVS Model

Figure 5.2. Outputs of Agent Based and DEVS models.

that the counterpart DEVS model is very efficient when compared with the agent-based model. This is because, a DEVS model is inherently more efficient in simulating systems which have spatio-temporal heterogeneity in their activity behaviors.

Table 5.3. Execution times for the agent-based model and its counterpart DEVS model.

Simulation Time	Agent-Based Implementation	DEVS Implementation
5 years	2 hours	1.2 minutes
25 years	10 hours	6 minutes
100 years	40 hours(extrapolated)	24 minutes

5.2 Description of Coarse and Fine Models

Figure 4.1 also shows the nine possible combinations of the constituents of the valley fever model. These nine different combinations can be arranged into a table. Table 5.4 shows these nine different combinations and their compatibility in relevance to the spread of the valley fever fungus.

In the table, coarse temperature refers to the temperature model which models the temperature using a Cosine wave as shown in figure 3.3. Fine temperature refers to the temperature model which models the temperature using the equation given in [4] whose fit is as shown in figure 3.8. Coarse moisture refers to the moisture model which computes the moisture at a depth of 20 cm using heat flow calculation. Fine moisture refers to the moisture model which models the moisture using equation 3.5

Table 5.4. Nine possible combinations of the constituents of the valley fever model.

	Fine Temp	Fine Moisture	Fine Ground
Coarse Temp	N/A	Incompatible	Compatible
Coarse Moisture	Incompatible	N/A	Compatible
Coarse Ground	Compatible	Compatible	N/A

whose fit is as shown in figure 3.9. Coarse ground refers to the ground model which has binary favorableness. That is, either the ground is favorable to the growth of the valley fever fungus or it is not favorable for the growth of the valley fever fungus. Fine ground refers to the ground model which has variable degree of favorableness. A picture of this ground model is as shown in figure 3.2

The fine temperature model and the fine moisture model are inter-linked to each other. For each cell in the cellspace, the diffusion coefficient κ of the temperature model is dependent on the moisture for that cell which is computed by 3.5. This relation is given by equation 3.3. Therefore one model is incompatible without the other.

In a similar manner, the coarse temperature model and the coarse moisture model are inter-linked with each other. In order to calculate the amount of moisture that is present at a depth of 20 cm at any given time, the temperature at two depths is calculated and the amount of water that accounts for that temperature difference is subtracted from the amount of water which is already present at that depth due to rainfall.

The combination (Coarse temperature,Fine temperature) is not applicable as only one of the temperature models must be used.

The combination (Coarse temperature, Fine moisture) is incompatible as temperature and moisture are inter-dependent.

The combination (Coarse temperature, Fine ground) is compatible as they are independent of each other.

The combination (Coarse moisture, Fine temperature) is incompatible as temperature and moisture are inter-dependent.

The combination (Coarse moisture, Fine moisture) is not applicable as only one of the moisture models must be used.

The combination (Coarse moisture, Fine ground) is compatible as they are independent of each other.

Table 5.5. Run times for the possible combinations for the valley fever model.

Moisture and Temp Models	Ground Model	Time Taken
Coarse	200 by 200 Coarse	6 minutes
Fine	200 by 200 Coarse	3 hours
Coarse	200 by 200 fine	8.5 hours
Coarse	1500 by 1500 fine	2.5 days
Fine	1500 by 1500 fine	75 days

The combination (Coarse ground, Fine temperature) is compatible as they are independent of each other.

The combination (Coarse ground, Fine moisture) is compatible as they are independent of each other.

The combination (Coarse ground, Fine ground) is not applicable as only one of the ground models must be used.

5.3 Execution Times of Coarse and Fine Models

Table 5.5 shows the run times(run times have been extrapolated where it is unreasonable to actually run the simulation) for the possible combinations for the valley fever model. These combinations were run on a machine with dual core AMD Athlon Processor (3.0 GHz) and 6.0 GB of RAM. The ADEVS [22] package(version 2.0.5.4) was used for running the valley fever model.

As it can be seen from the table, if the user is interested in a high level or qualitative view of the spread of the valley fever fungus, he should just use the coarse combinations in order to see the results quickly.

On the other hand, if he is interested in a low level or quantitative view of the spread of the valley fever fungus, he should use the fine combinations in order to get the detailed results.

In chapter 6, we will compare the base and lumped models from the point of view of parameter morphism and self-organized criticality.

Chapter 6

Use of the multilevel modeling methodology to study self-organized criticality of the valley fever model

This chapter has two parts. The first part gives a brief introduction to the concept of self-organized criticality [2, 1, 6]. The first part also shows the results of simulating the valley fever model using the multilevel modeling methodology. As mentioned before, the agent-based implementation of the valley fever model uses the coarse moisture and temperature models. In order to faithfully reproduce the same, we included the same coarse versions of the moisture and temperature models into the lumped version of the valley fever model. According to expert judgement of domain experts such as Dr. Gettings, the use of the coarse versions of the moisture and temperature models results in the formation of far more new sites(dormant+sporing sites) in the valley fever model. This observation can be made by looking at the results shown in this part which show that the lumped model version of the valley fever model produces significant number of new sites(dormant+sporing sites) when compared with the corresponding base version of the valley fever model which uses the fine versions of the moisture and temperature models. For each of the results, the number of new sites(dormant+sporing sites) is shown as a function of time. This is useful because it helps us in understanding the weather patterns which cause an outbreak in the valley fever fungus. This information can then be used in predicting what weather patterns to look for in the future that might result in an outbreak of the valley fever fungus. The second part discusses about parameter morphism and discusses about future work that needs to be done in this area.

6.1 Self-Organized Criticality

Self-Organized Criticality also known as SOC [2, 1] is an attempt to explain the occurrence of complex phenomena. The valley fever model is an example of a complex system as the effect of each factor which affects the growth of the fungus is known individually, but, its hard to predict the combined effect that these factors would have on the growth of the valley fever fungus. It is an example of a system which exhibits emergent behavior where simple rules in a system give rise to complex behavior [6].

The following are the characteristics of an SOC system [6]:

1. The components in the system are governed by simple rules.
2. Threshold exists within the system.
3. Pressure builds in the system until it exceeds the threshold.
4. Small agitations in the system can lead to system effects called avalanches.

The next section shows the application of the multilevel modeling methodology to study self-organized criticality.

6.1.1 Multilevel Modeling Methodology to Study Self-Organized Criticality

This section shows several results of running the lumped and base versions of the valley fever model by varying the parameters which affect the spread of the valley fever fungus. It also shows results which validate the use of the new moisture and temperature models by verifying that their use reduces the formation of new sites in the valley fever model.

The parameters that affect the growth of the fungus are:

1. Number of patches to blow the wind (denoted by wjmax).

2. Azimuth of the trade wind (denoted by tradeWindAz).

3. Azimuth range for the wind (denoted by windAZRange).

4. Maximum tolerable temperature for the fungus (denoted by tempMax).

5. Minimum moisture required for the survival of the fungus (denoted by moistMin).

6. Time taken to form a new site (denoted by SiteMin).

The distance that the wind can blow spores to is directly proportional to the value of the parameter wjmax. As a result of this, other factors being favorable, a high value of wjmax facilitates the formation of new sites.

The parameter tradeWindAz denotes the direction in which the wind is blowing. A value of 0 ° denotes that the wind is blowing towards North, a value of 90 ° denotes that the wind is blowing towards the East, a value of 180 ° denotes that the wind is blowing towards South and a value of 270 ° denotes that the wind is blowing towards the West. The direction of the wind plays a significant role in determining the spread of the fungus as the wind is the main carrier of spores in the valley fever model.

The parameter windAZRange denotes the angle (or the width) in which the wind can blow. This also plays a significant role in determining the spread of the valley fever fungus.

The parameter tempMax denotes the maximum tolerable temperature for the survival of the fungus. That is, a soil temperature that exceeds this value is unfavorable for the growth of the valley fever fungus.

The parameter moistMin denotes the minimum amount of moisture that is required for the survival of the fungus. That is, the fungus cannot survive in a soil whose moisture content is less than moistMin.

The parameter SiteMin denotes the time taken by the fungus to form a new site. The higher this value is, the lower is the spread of the valley fever fungus.

In all figures, a White color denotes unfavorable ground, a Yellow color denotes favorable ground, a Green color denotes that the patch is in a colonized phase, a Red color denotes that the patch is in a sporing phase and a Purple color denotes that the patch is in a dormant phase. The ADEVS [22] package(version 2.0.5.4) was used for running the valley fever model.

6.2 Binary Favorableness

The ground model with binary favorableness shown in figure 6.1 is used as input for both the base and lumped models. As will be shown, the use of the base model has its pros and cons. The main advantage is that the valley fever model is mathematically more accurate and is based on temperature and moisture observations. As a result, the base model is more realistic. This is reflected in the few number of new sites that are formed when using the base model. The main disadvantage is the run time. For a typical grid size of 200 × 200, the base model takes about 3 hours to run, whereas the lumped model takes about 6 minutes to run.

The parameters which were described at the beginning of this section were varied in six different ways most of which were taken from [10]. The first case is a test case. Table 6.1 shows the values of all the parameters for each of these test cases.

Table 6.1. Parameters for a test run of the valley fever model.

Parameter	1	2	3	4	5	6
wjmax	5.0	1.0	1.0	5.0	5.0	1.0
tradeWindAz	0	271	89	0	0	0
windAZRange	89	89	89	89	60	89
tempMax	40	30	25	40	40	40
moistMin(inches)	0.5	0.4	0.5	0.2	0.2	0.2
SiteMin(days)	9	9	13	9	9	9

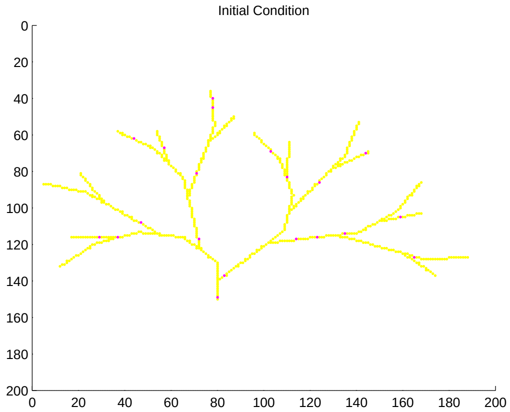


Figure 6.1. Input to the valley fever model.

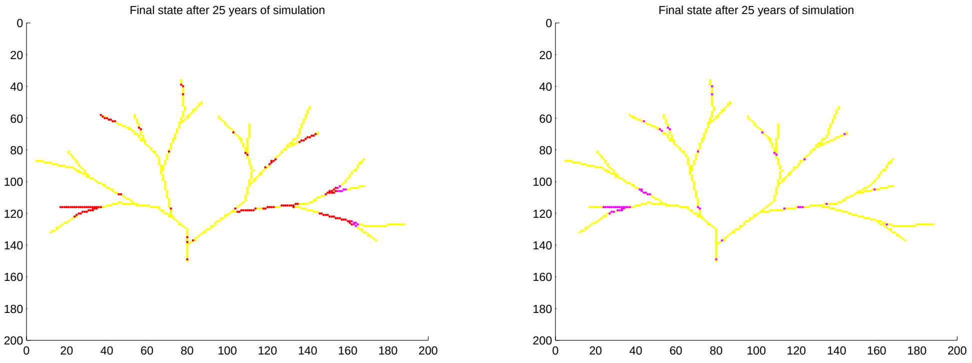
Figure 6.1 shows the input that was given to the base and lumped versions of the valley fever model.

Figures 6.3, 6.4, 6.5, 6.6, 6.7 show the results of simulating the basic version (with binary favorableness) of the valley fever model for 25 years when the parameters were chosen according to column 1 of table 6.1. Figure 6.2 shows the outputs of the models in this case.

Figure 6.5 shows the total number of new sites of the valley fever model for 25 years when the parameters were chosen according to column 1 of table 6.1. This figure is an example of the self-organized criticality behavior in the valley fever model. Similar behavior can be seen in other test cases of table 6.1.

Figures 6.9, 6.10, 6.11, 6.12, 6.13 show the results of simulating the basic version (with binary favorableness) of the valley fever model for 25 years when the parameters were chosen according to column 2 of table 6.1. These parameters were taken from [10]. Figure 6.8 shows the output of the model in this case.

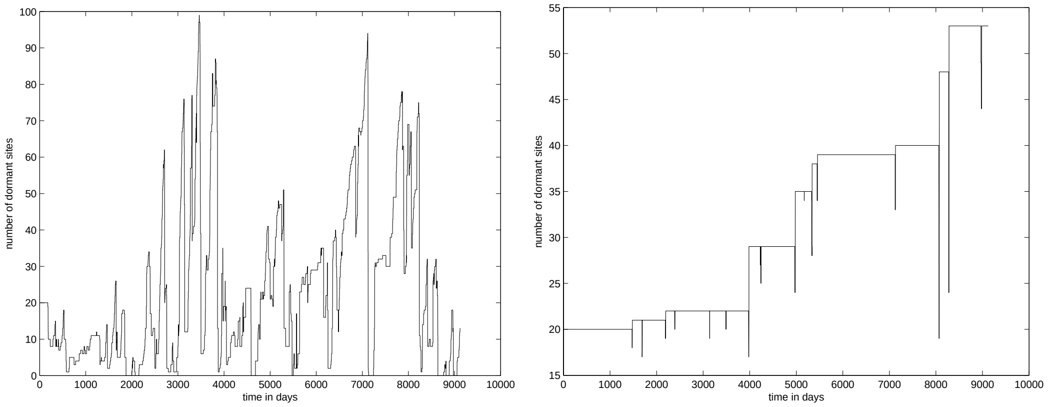
Figures 6.15, 6.16, 6.17, 6.18, 6.19 show the results of simulating the basic version (with binary favorableness) of the valley fever model for 25 years when the parameters were chosen according to column 3 of table 6.1. These parameters were taken from [10]. Figure 6.14 shows the output of the models in this case.



(a) Lumped Model

(b) Base Model

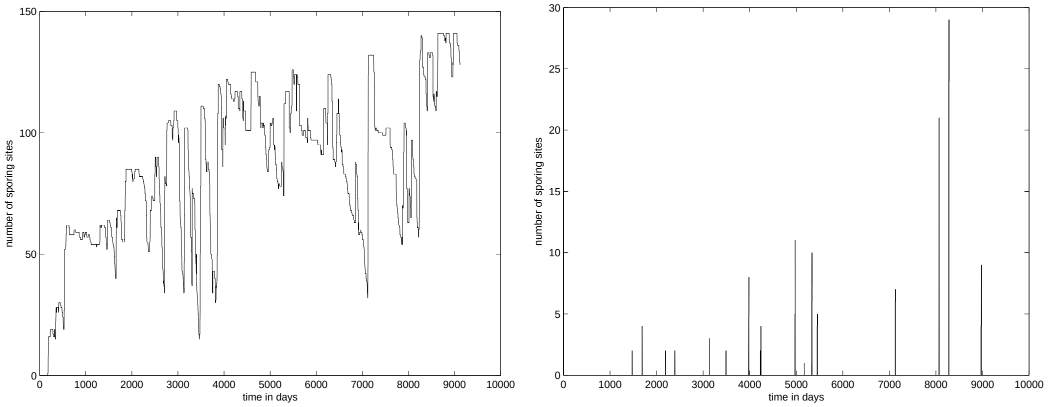
FIGURE 6.2. Results of test case for the lumped and base valley fever models.



(a) Lumped Model

(b) Base Model

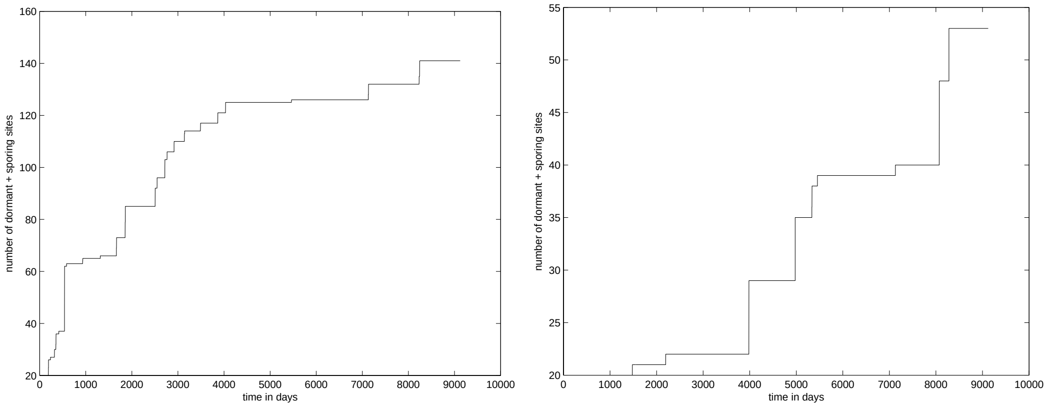
FIGURE 6.3. Number of dormant sites as a function of time.



(a) Lumped Model

(b) Base Model

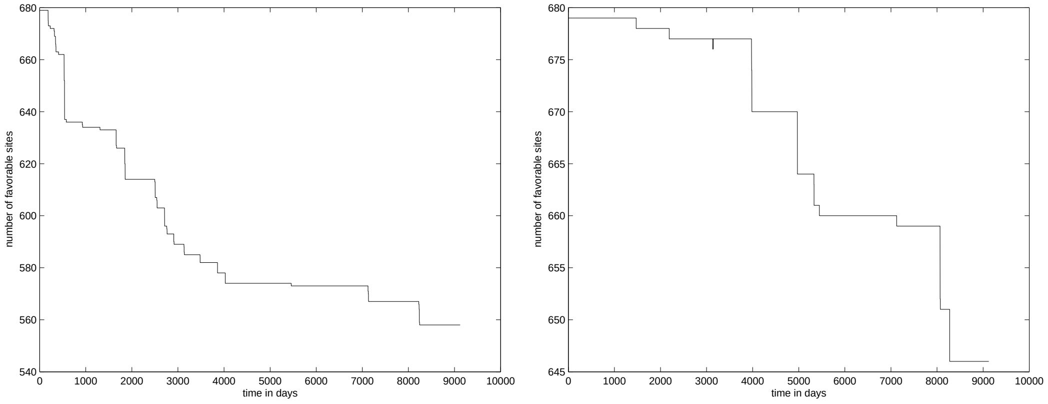
FIGURE 6.4. Number of sporing sites as a function of time.



(a) Lumped Model

(b) Base Model

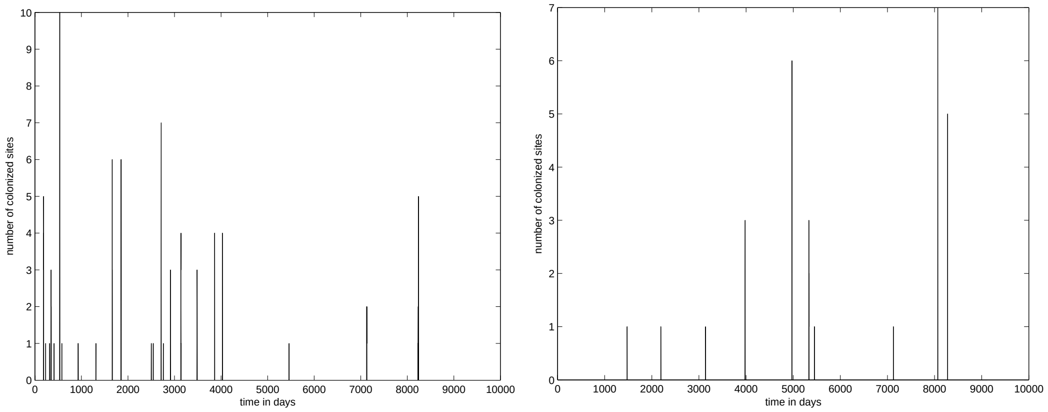
FIGURE 6.5. Number of dormant and sporing sites as a function of time.



(a) Lumped Model

(b) Base Model

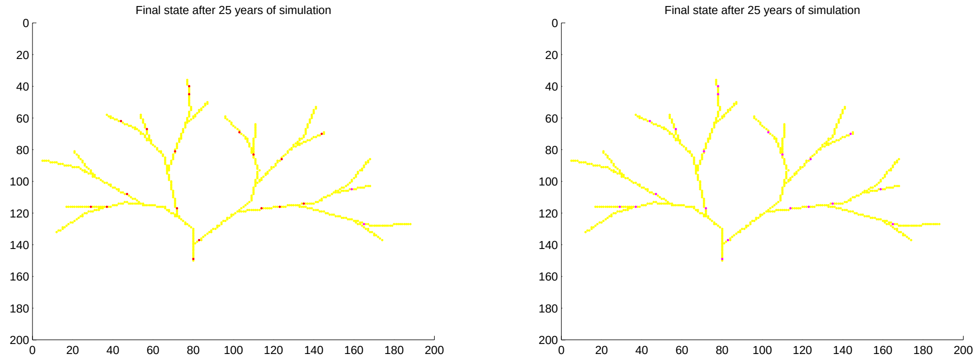
FIGURE 6.6. Number of favorable sites as a function of time.



(a) Lumped Model

(b) Base Model

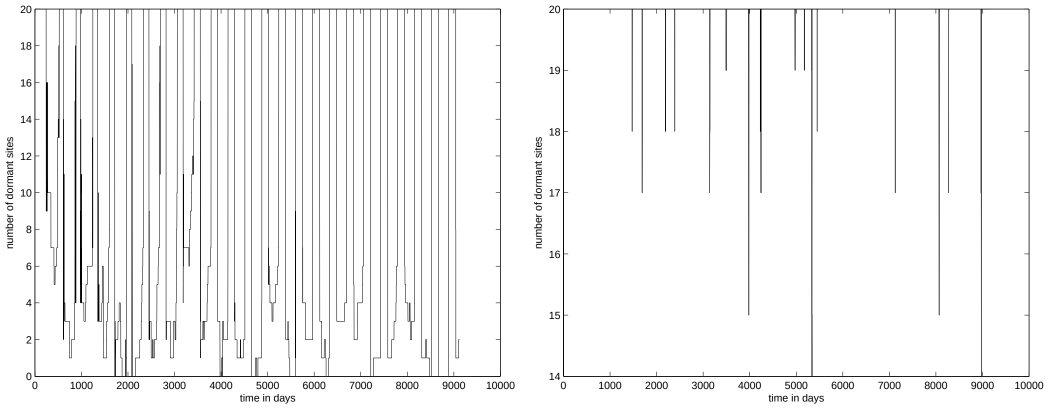
FIGURE 6.7. Number of colonized sites as a function of time.



(a) Lumped Model

(b) Base Model

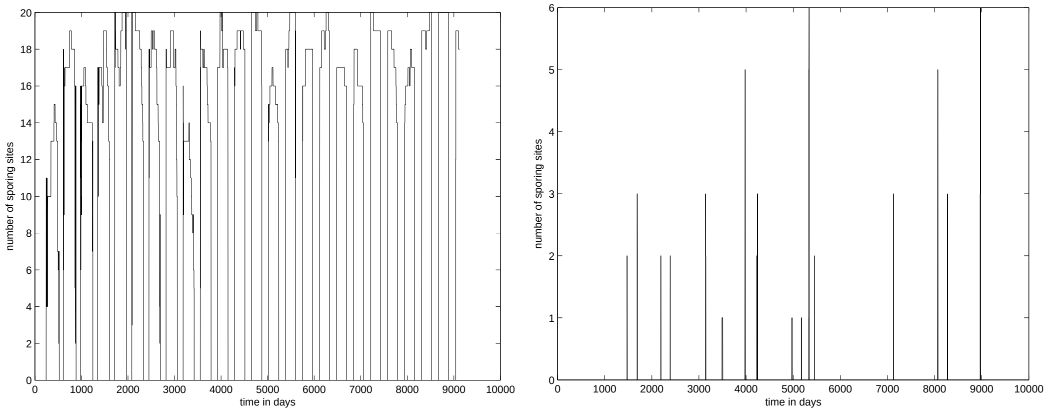
FIGURE 6.8. Results of lumped and base models for the parameters in column 2 of table 6.1.



(a) Lumped Model

(b) Base Model

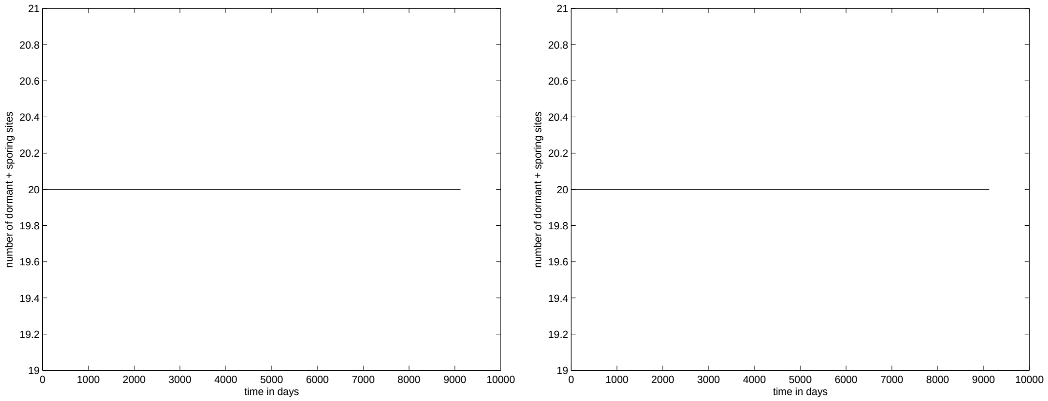
FIGURE 6.9. Number of dormant sites as a function of time.



(a) Lumped Model

(b) Base Model

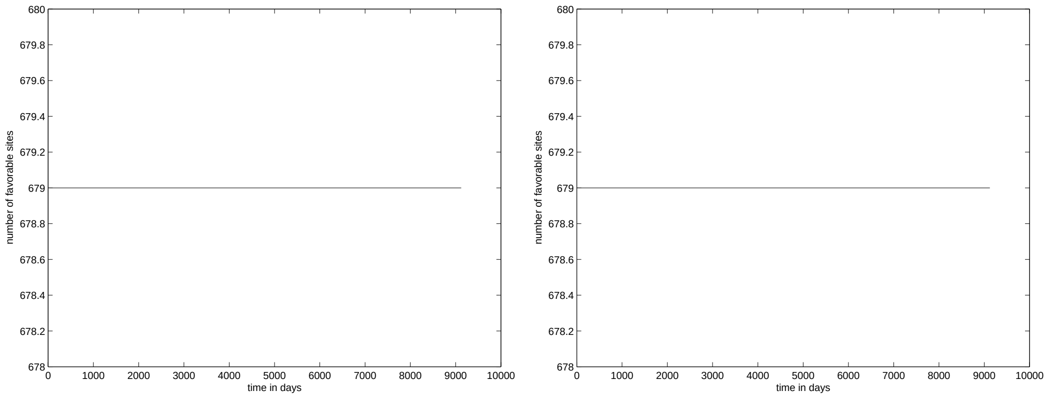
Figure 6.10. Number of sporing sites as a function of time.



(a) Lumped Model

(b) Base Model

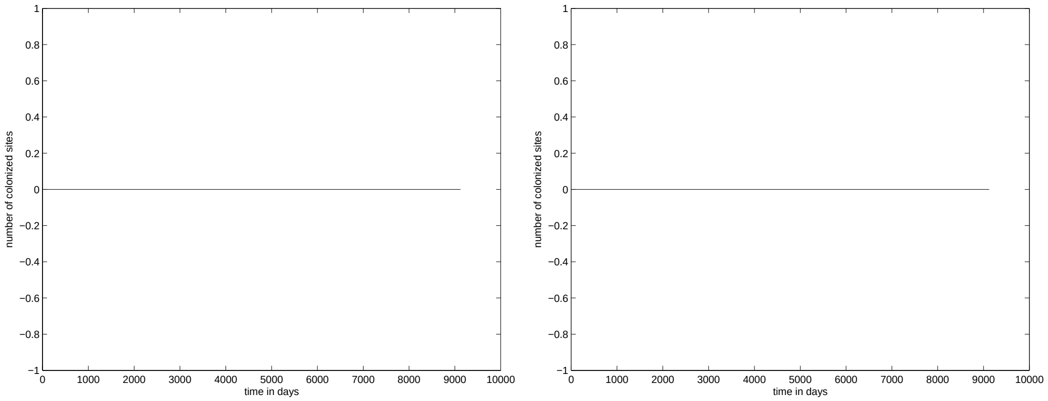
Figure 6.11. Number of dormant and sporing sites as a function of time.



(a) Lumped Model

(b) Base Model

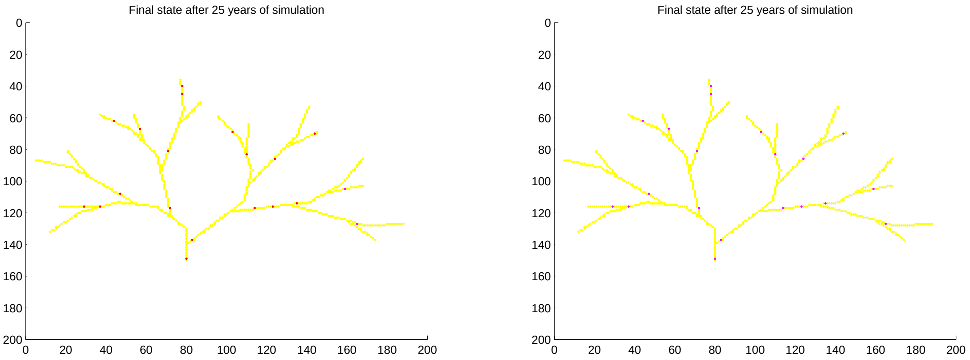
Figure 6.12. Number of favorable sites as a function of time.



(a) Lumped Model

(b) Base Model

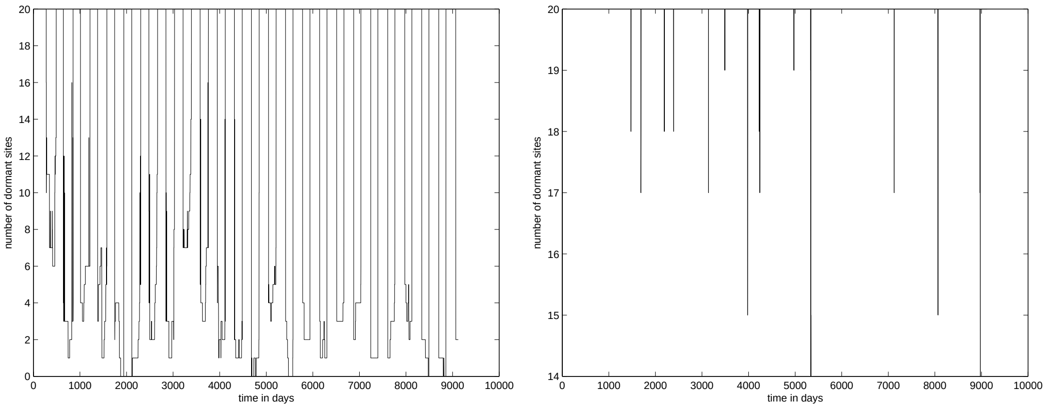
Figure 6.13. Number of colonized sites as a function of time.



(a) Lumped Model

(b) Base Model

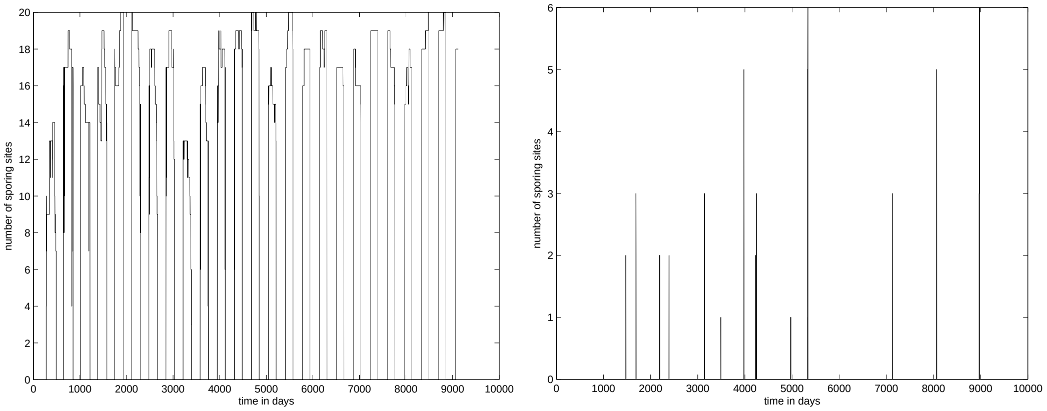
Figure 6.14. Results of lumped and base models for the parameters in column 3 of table 6.1.



(a) Lumped Model

(b) Base Model

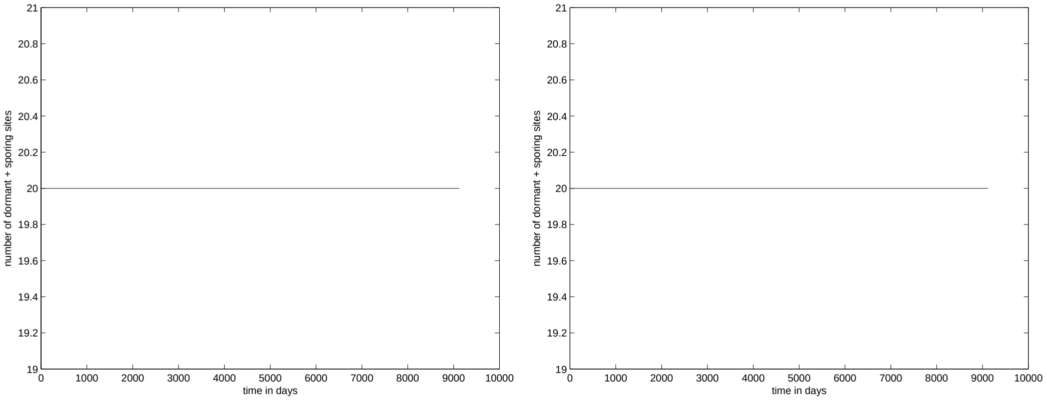
Figure 6.15. Number of dormant sites as a function of time.



(a) Lumped Model

(b) Base Model

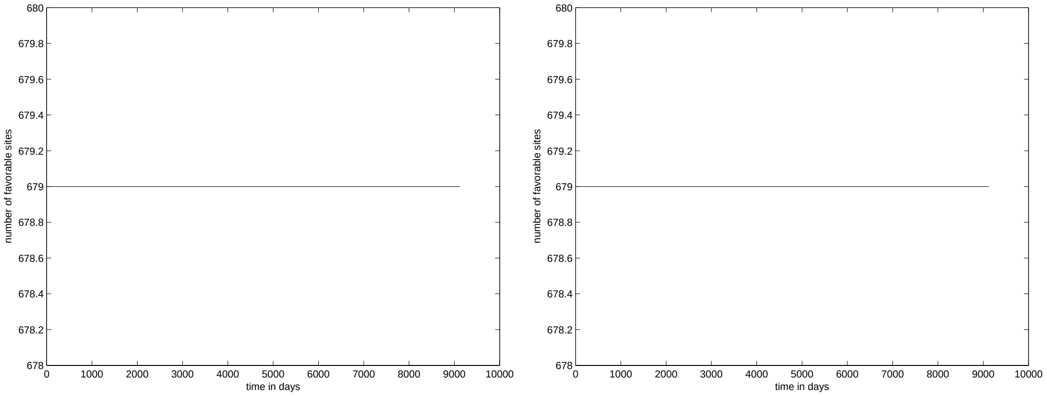
Figure 6.16. Number of sporing sites as a function of time.



(a) Lumped Model

(b) Base Model

Figure 6.17. Number of dormant and sporing sites as a function of time.

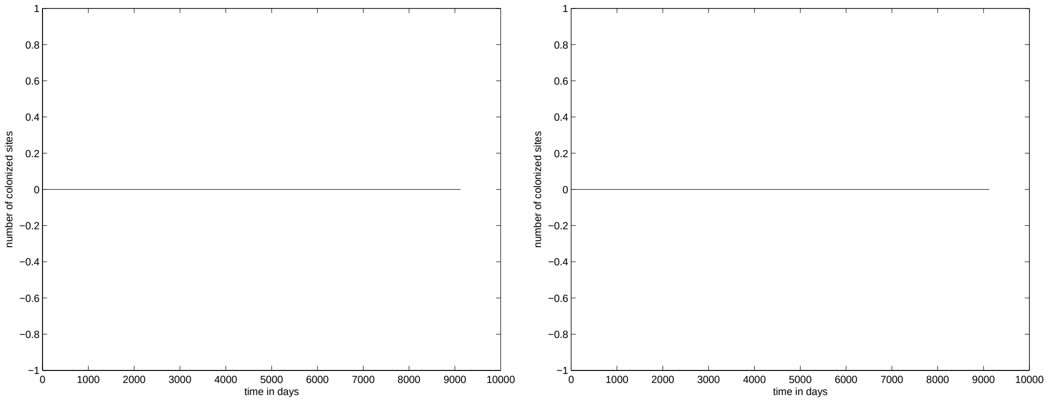


(a) Lumped Model

(b) Base Model

Figure 6.18. Number of favorable sites as a function of time.

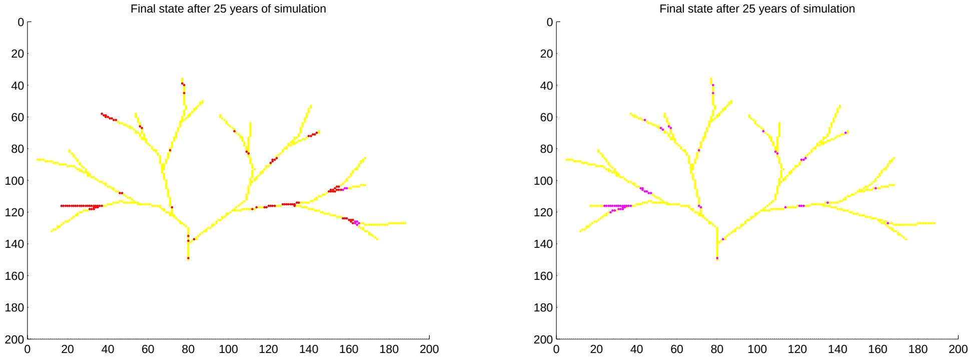
Figures 6.21, 6.22, 6.23, 6.24, 6.25 show the results of simulating the basic version (with binary favorableness) of the valley fever model for 25 years when the parameters were chosen according to column 4 of table 6.1. These parameters were taken from [10]. Figure 6.20 shows the output of the models in this case.



(a) Lumped Model

(b) Base Model

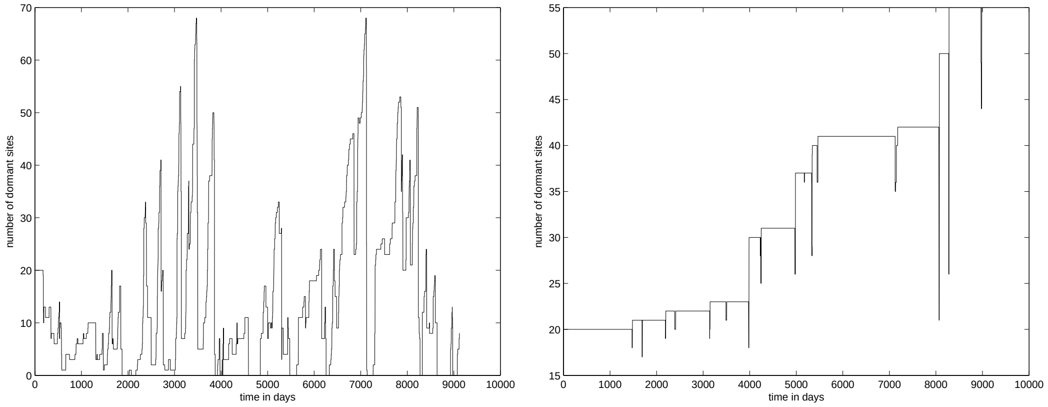
FIGURE 6.19. Number of colonized sites as a function of time.



(a) Lumped Model

(b) Base Model

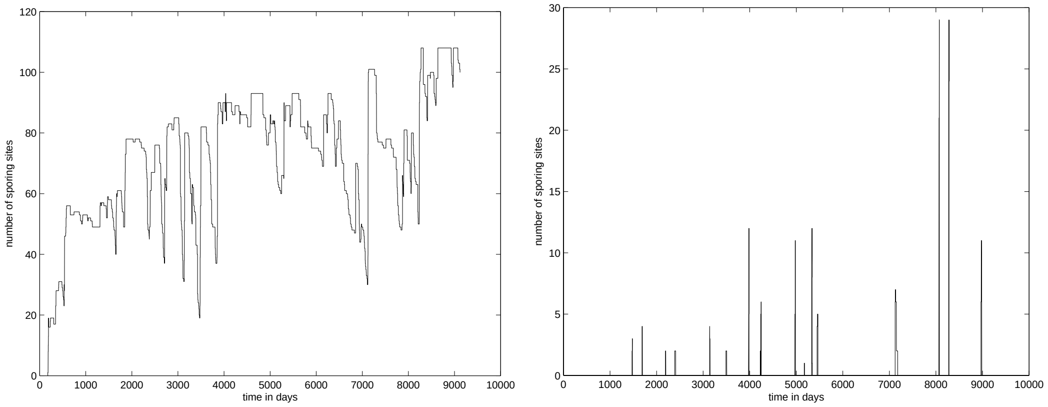
FIGURE 6.20. Results of the lumped and base models for the parameters in column 4 of table 6.1.



(a) Lumped Model

(b) Base Model

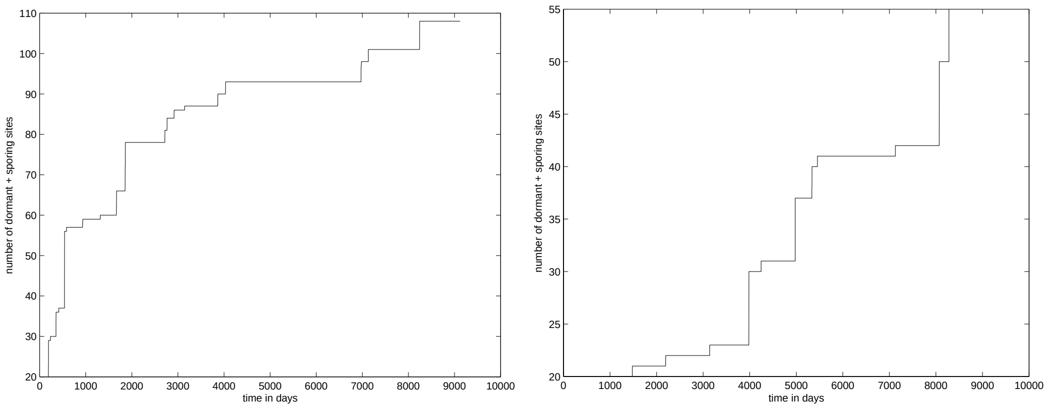
Figure 6.21. Number of dormant sites as a function of time.



(a) Lumped Model

(b) Base Model

Figure 6.22. Number of sporing sites as a function of time.



(a) Lumped Model

(b) Base Model

FIGURE 6.23. Number of dormant and sporing sites as a function of time.

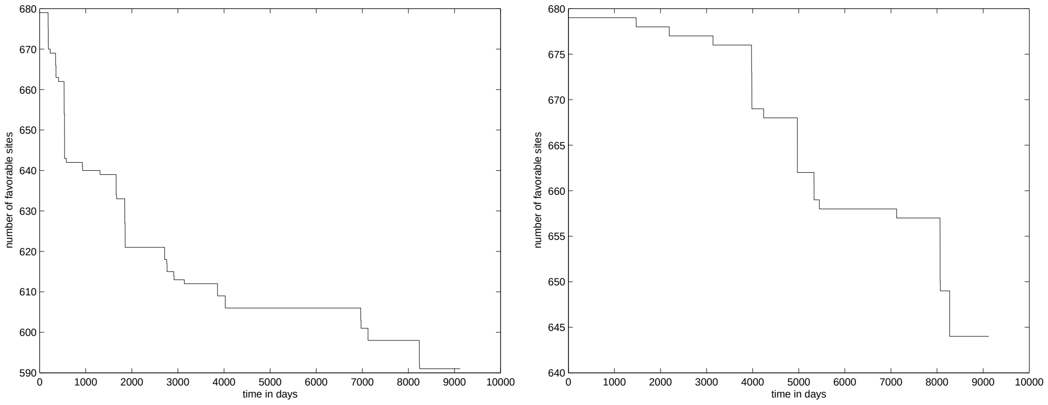
Figures 6.27, 6.28, 6.29, 6.30, 6.31 show the results of simulating the basic version (with binary favorableness) of the valley fever model for 25 years when the parameters were chosen according to column 5 of table 6.1. These parameters were taken from [10]. Figure 6.26 shows the output of the models in this case.

Figures 6.33, 6.34, 6.35, 6.36, 6.37 show the results of simulating the basic version (with binary favorableness) of the valley fever model for 25 years when the parameters were chosen according to column 6 of table 6.1. These parameters were taken from [10]. Figure 6.32 shows the output of the models in this case.

Table 6.2 summarizes the results of the testcases shown in 6.1.

As can be inferred from table 6.2, the use of the Base model significantly reduces the formation of new sites. It can also be noticed that after 25 years of simulation, the base model contains only sites which are dormant, whereas the lumped model contains a large number of sporing sites (which can serve as a source of formation for new sites). This validates the use of the new mathematical models for temperature and moisture which reduce the formation of new sites in the valley fever model.

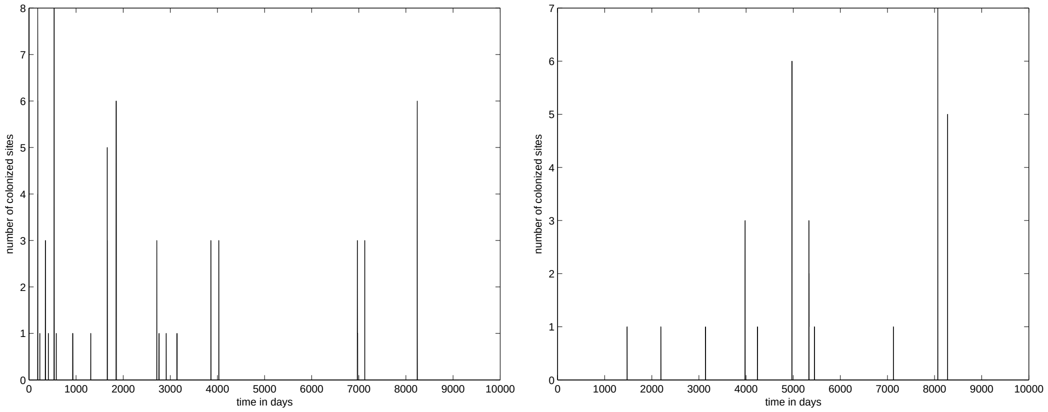
But, as noted before, this advantage of accuracy comes with a cost. The time taken



(a) Lumped Model

(b) Base Model

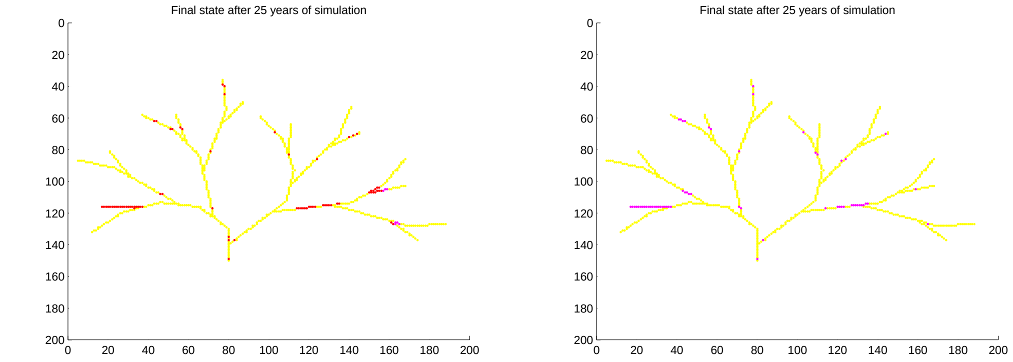
Figure 6.24. Number of favorable sites as a function of time.



(a) Lumped Model

(b) Base Model

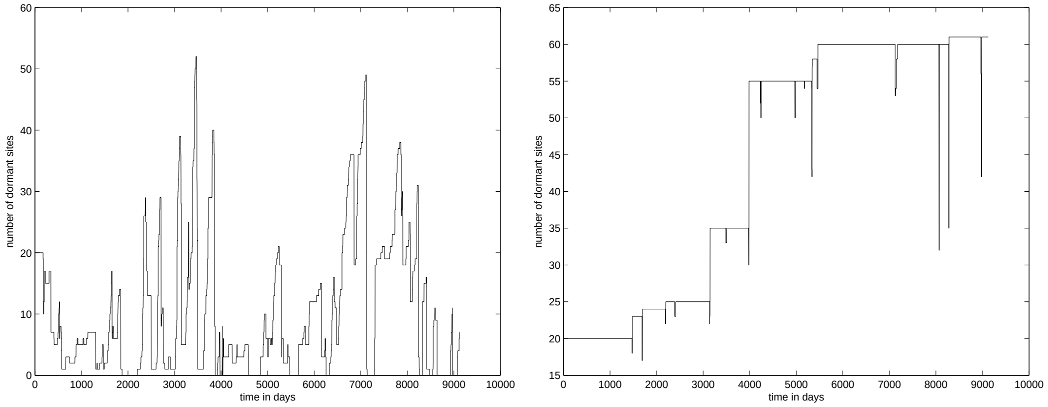
Figure 6.25. Number of colonized sites as a function of time.



(a) Lumped Model

(b) Base Model

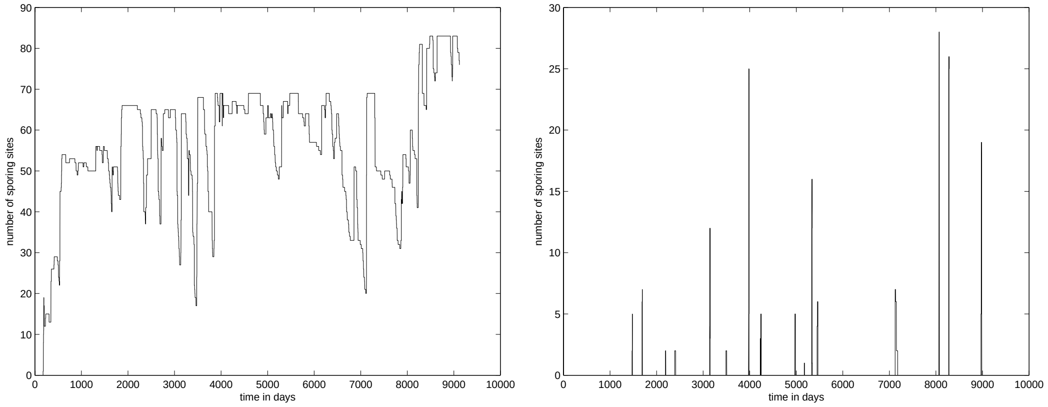
Figure 6.26. Results of the lumped and base models for the parameters in column 5 of table 6.1.



(a) Lumped Model

(b) Base Model

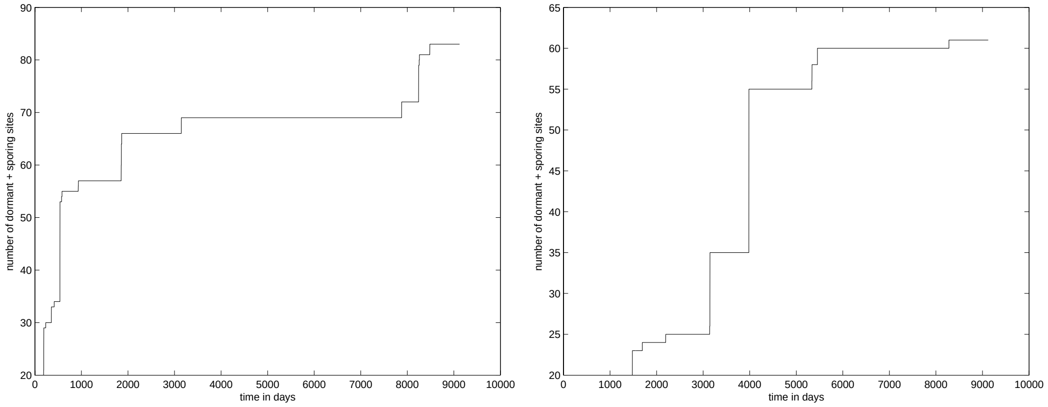
Figure 6.27. Number of dormant sites as a function of time.



(a) Lumped Model

(b) Base Model

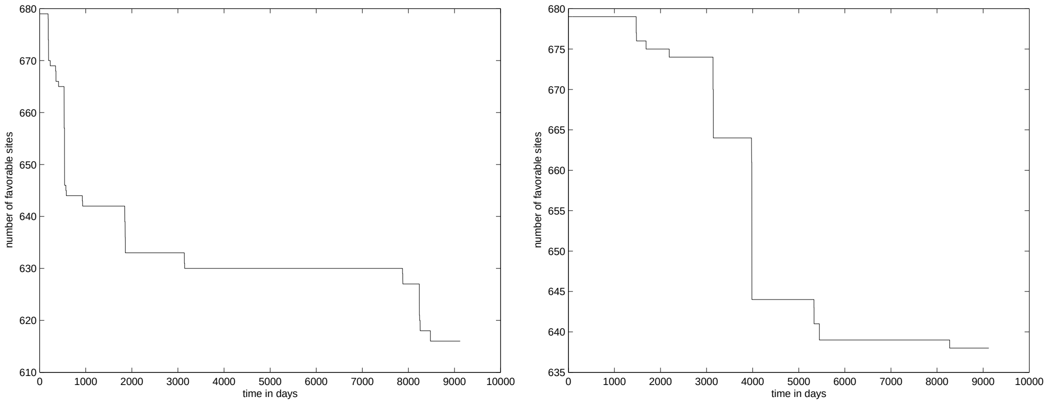
Figure 6.28. Number of sporing sites as a function of time.



(a) Lumped Model

(b) Base Model

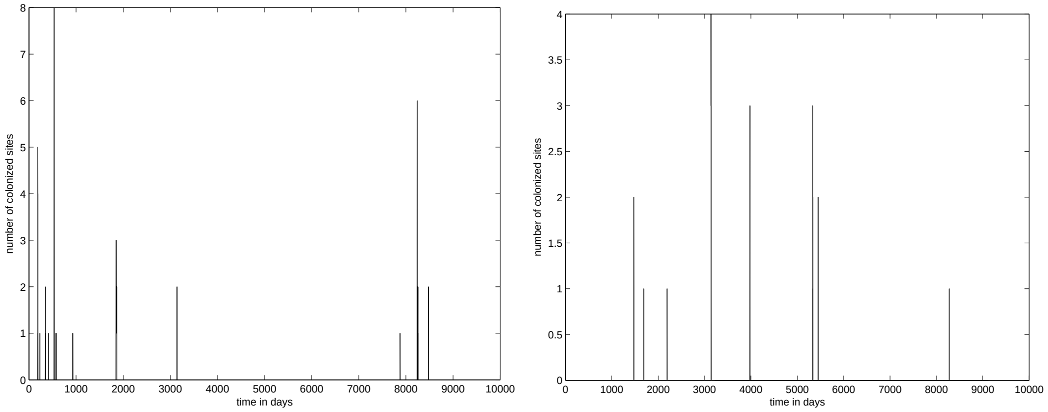
Figure 6.29. Number of dormant and sporing sites as a function of time.



(a) Lumped Model

(b) Base Model

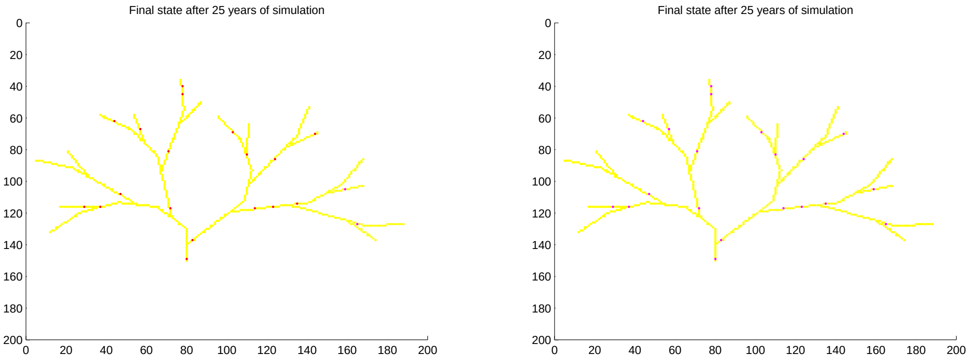
FIGURE 6.30. Number of favorable sites as a function of time.



(a) Lumped Model

(b) Base Model

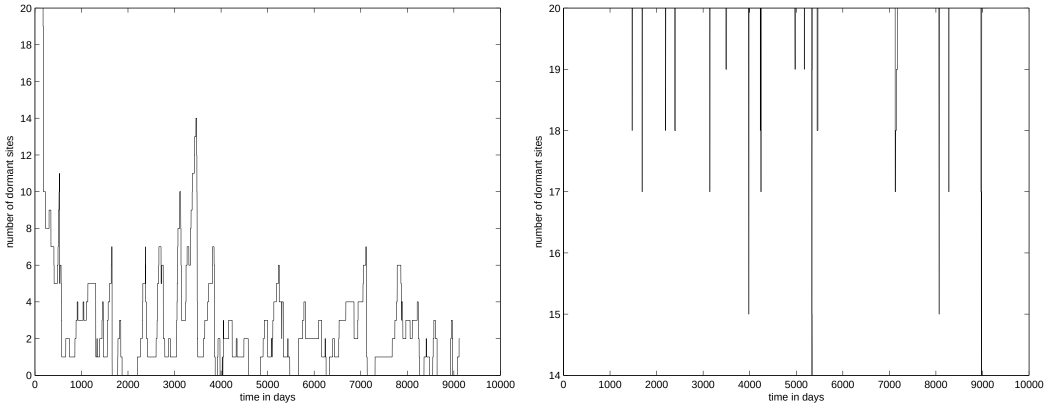
FIGURE 6.31. Number of colonized sites as a function of time.



(a) Lumped Model

(b) Base Model

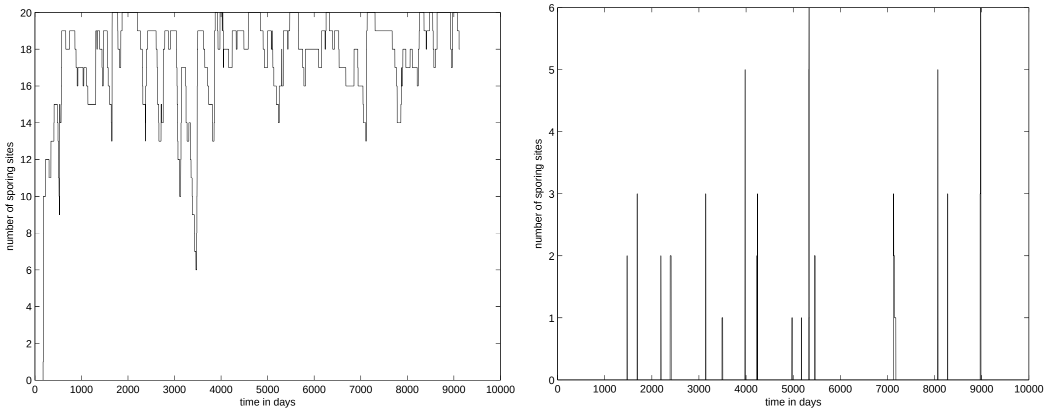
Figure 6.32. Results of the Lumped and Base models for the parameters in column 6 of table 6.1.



(a) Lumped Model

(b) Base Model

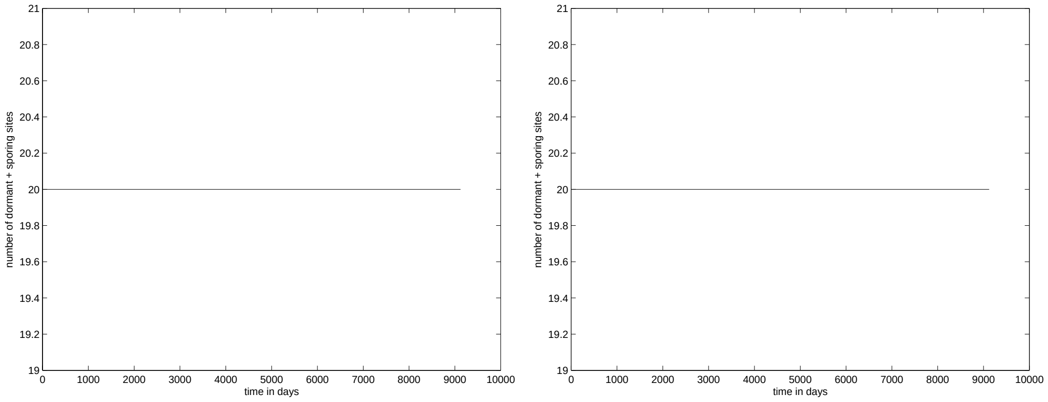
Figure 6.33. Number of dormant sites as a function of time.



(a) Lumped Model

(b) Base Model

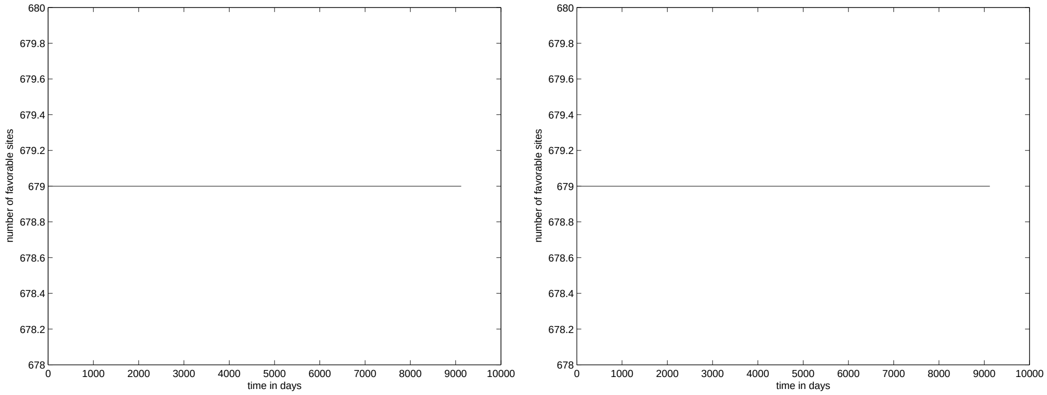
Figure 6.34. Number of sporing sites as a function of time.



(a) Lumped Model

(b) Base Model

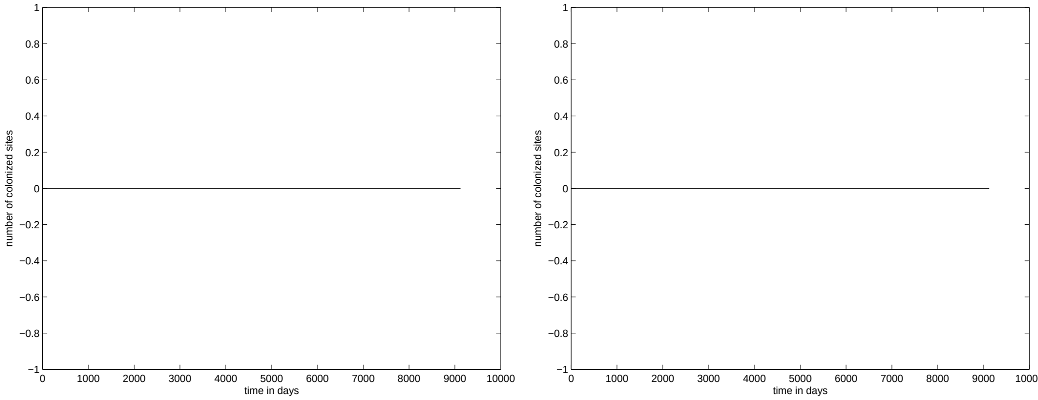
Figure 6.35. Number of dormant and sporing sites as a function of time.



(a) Lumped Model

(b) Base Model

Figure 6.36. Number of favorable sites as a function of time.



(a) Lumped Model

(b) Base Model

Figure 6.37. Number of colonized sites as a function of time.

Table 6.2. Results of the test cases in table 6.1 for the Lumped and Base models.

Test Case	Lumped (Sporing/Dormant/Total)	Base (Sporing/Dormant/Total)
1	128, 13, 141	0, 53, 53
2	18, 2, 20	0, 20, 20
3	18, 2, 20	0, 20, 20
4	100, 8, 108	0, 55, 55
5	76, 7, 83	0, 61, 61
6	18, 2, 20	0, 20, 20

Table 6.3. Run times of the test cases in table 6.1 for the Lumped and Base models.

Case	Lumped Model Run Time(in Hours)	Base Model Run Time(in Hours)
1	0.102	3.11
2	0.104	3.10
3	0.105	3.2
4	0.107	3.21
5	0.107	3.10
6	0.101	3.12
7	0.104	3.10

to run the base model is about 30 times the time taken to run the corresponding lumped model. Table 6.3 summarizes the results of running the simulation on a machine with a dual core AMD Athlon Processor (3.0 GHz) and 6 GB of RAM for each of the test cases in table 6.1.

6.3 Variable Favorableness

This section shows some results of running the lumped version of the valley fever model along with the fine resolution(variable favorableness) ground model.

Figure 6.39 shows the result of running the lumped version of the valley fever model for 25 years when the input is given as in figure 6.38. The dormant sites were chosen at the sites which were most favorable for the growth of the valley fever fungus. This simulation took 2.5 days when run on a machine with a dual core Athlon Processor (3.0 GHz) and 6 GB of RAM. All the parameters except SiteMin were chosen according to column 4 of table 6.1. SiteMin for each site was varied inversely as the favorableness of the ground at that site.

Figures 6.40, 6.41, 6.42, 6.44, 6.43 show the results of simulating the valley fever model for 25 years.

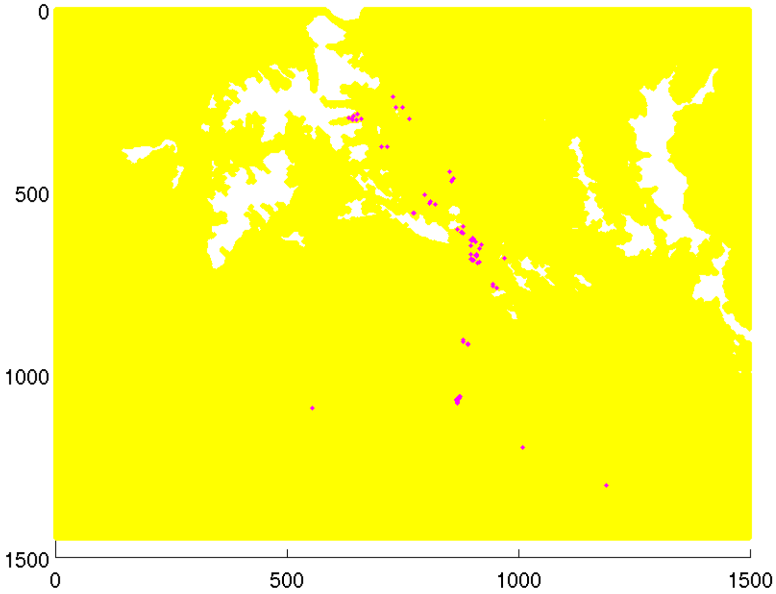


Figure 6.38. the input to the valley fever model with a ground model of variable favorableness.

Figure 6.45 shows the result of running the lumped version of the valley fever model for 25 years when the input is given as in figure 6.38. The dormant sites were chosen at the sites which were most favorable for the growth of the valley fever fungus. This simulation took 2.5 days when run on a machine with a dual core Athlon Processor (3.0 GHz) and 6 GB of RAM. All the parameters except SiteMin and moistMin were chosen according to column 4 of table 6.1. SiteMin was varied inversely as the favorableness of the ground. A value of 0.4 was assigned to the

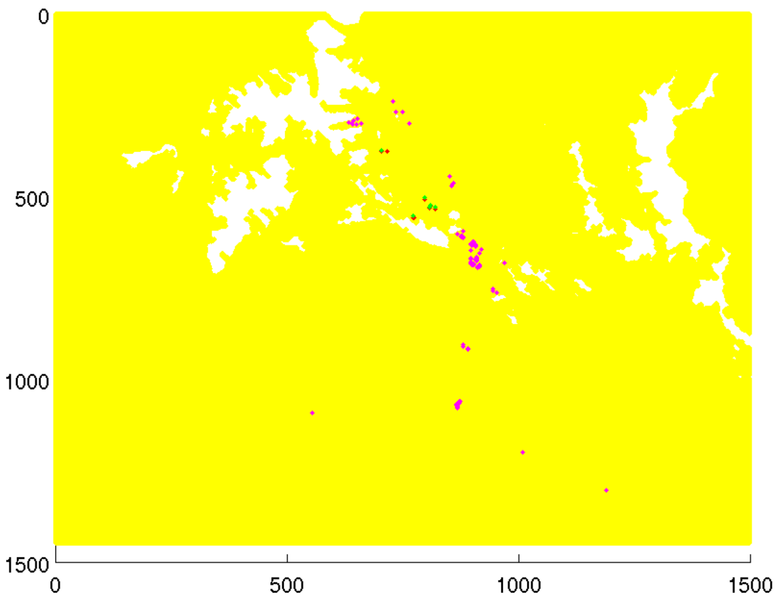


FIGURE 6.39. the output of the valley fever model with a ground model of variable favorableness after 25 years of simulation.

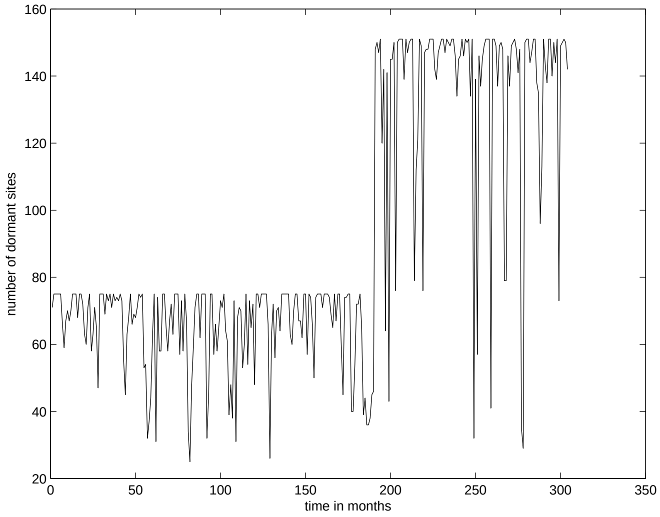


FIGURE 6.40. Number of dormant sites as a function of time.

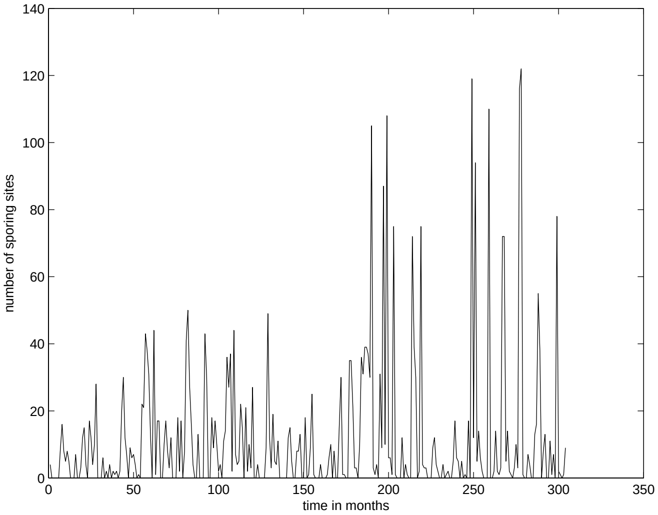


Figure 6.41. Number of sporing sites as a function of time.

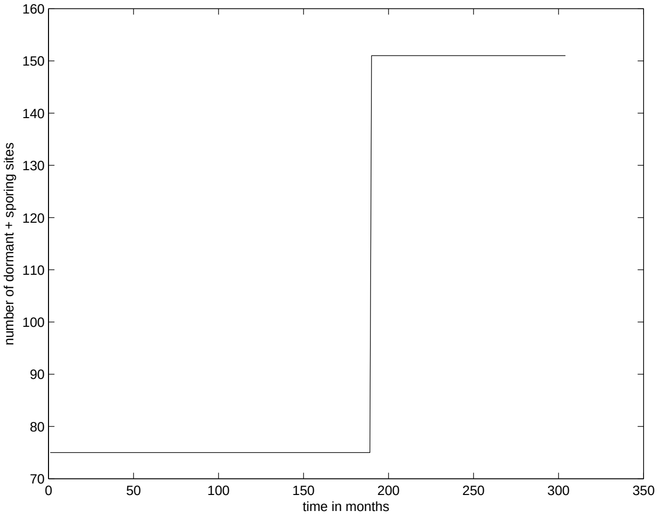


Figure 6.42. Number of dormant and sporing sites as a function of time.

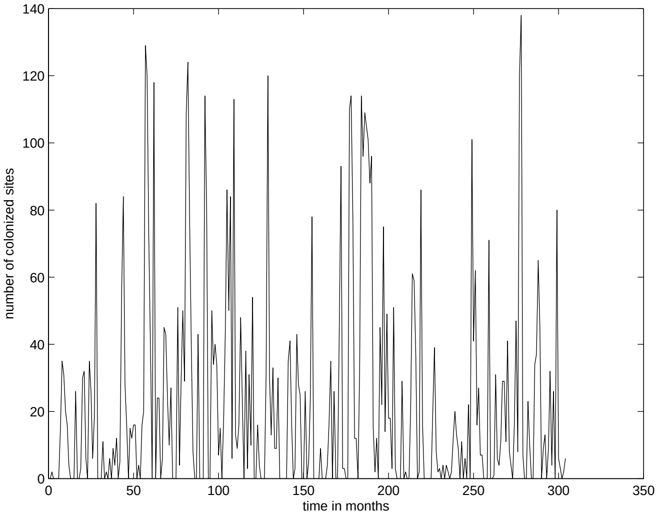


FIGURE 6.43. Number of colonized sites as a function of time.

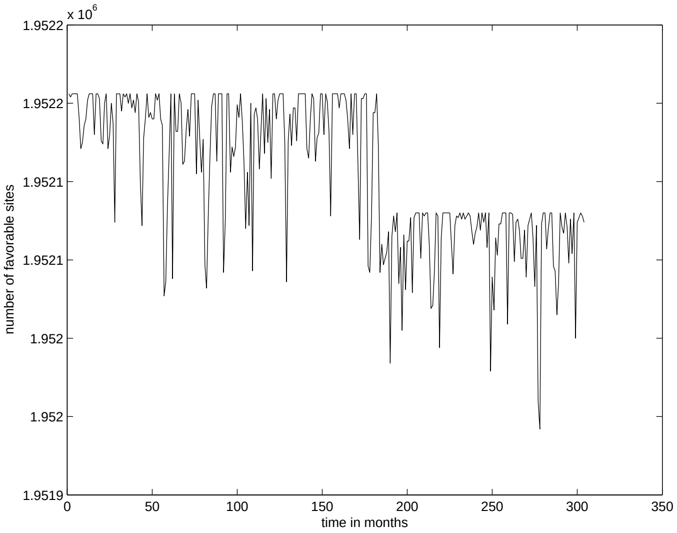


FIGURE 6.44. Number of favorable sites as a function of time.

Table 6.4. Result of running the lumped version of the valley fever model along with the ground model with variable favorableness.

Case	Input(dormant sites)	Output(dormant/sporing/colonized)
4	75	142, 9, 6
4(moistMin=0.4)	75	75,0,8

variable moistMin.

Figures 6.46, 6.47, 6.48, 6.49 and 6.50 show the results of simulating the valley fever model for 25 years.

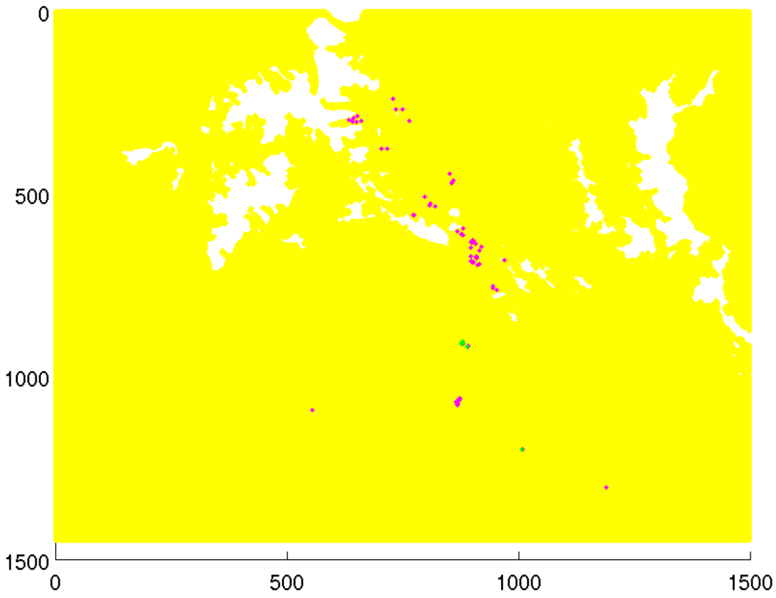


Figure 6.45. The output of the valley fever model with a ground model of variable favorableness after 25 years of simulation.

Table 6.4 summarizes the results of these two simulation.

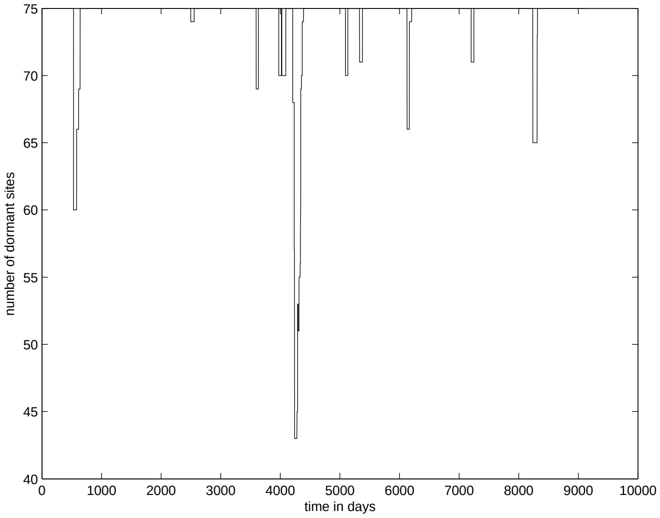


Figure 6.46. Number of dormant sites as a function of time.

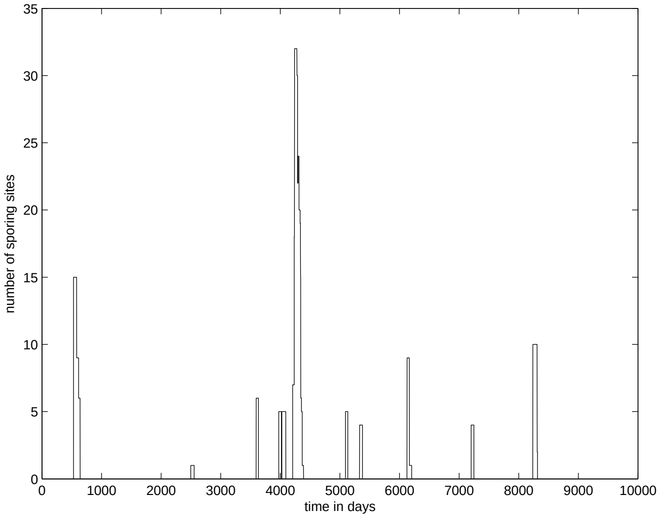


Figure 6.47. Number of sporing sites as a function of time.

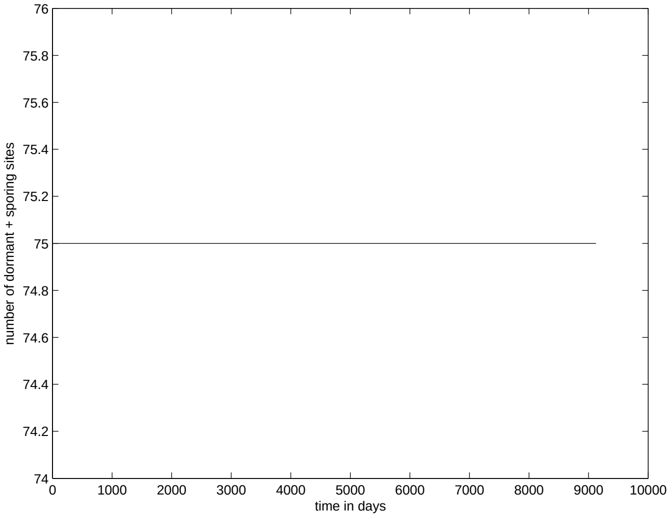


FIGURE 6.48. Number of dormant and sporing sites as a function of time.

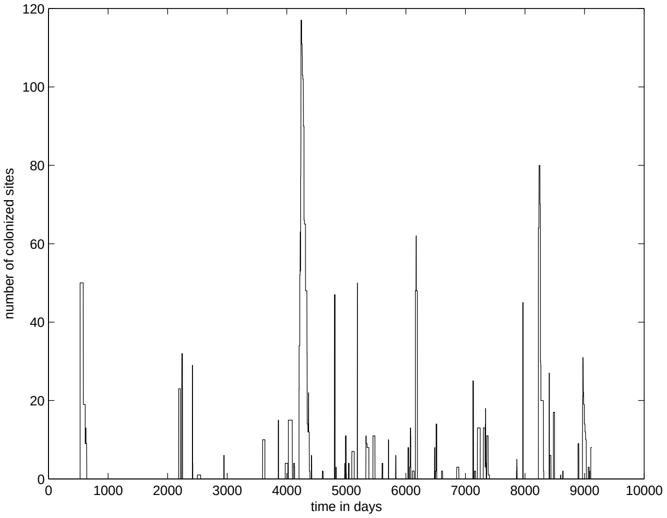


FIGURE 6.49. Number of colonized sites as a function of time.

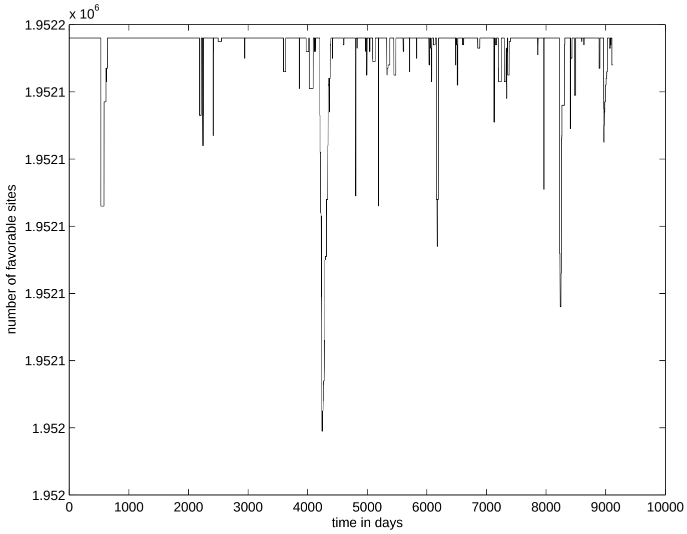


FIGURE 6.50. Number of favorable sites as a function of time.

6.4 Parameter Morphism

The goal of parameter morphism is to map the parameters of the base model to the parameters of the lumped model while preserving the behavior of the base model within a specified experimental frame. For the valley fever model, the experimental frame is the formation of new sites(dormant+sporing sites). The base version of the moisture model depends upon the rain events that have occurred in the past. The soil has a certain porosity and therefore the amount of moisture at a certain depth cannot exceed the total amount of rain that fell on the surface of the soil in the past. In order to account for this in the base version of the moisture model, the value of the computed moisture was compared with a predetermined threshold. If the computed moisture was greater than the threshold, the threshold was returned, otherwise the computed moisture value was returned.

In order to mimic similar behavior, we experimented with the lumped version of the moisture model(for the test case in column 1 of table 6.1) by setting a threshold limit for the computed moisture to see the effect of it on the spread of the valley fever fungus.

Table 6.5 shows the results of this experiment after simulating the lumped and base versions of the valley fever model for 25 years.

Table 6.5. Parameter morphism experiment.

Threshold(Moisture Model)	New sites(Lumped Model)	New Sites(Base Model)
0.8 inches	0	53
1.0 inches	0	53
No threshold	98	53
1.2 inches	267	53

From table 6.5, we can see that as the threshold for the lumped version of the moisture model is varied, the number of new sites formed after 25 years of simulation changes. The rate of the growth of the valley fever fungus is higher when the threshold value of the computed moisture is set to 1.2 inches when compared with the threshold value of 1.0 inches.

In this chapter, we have shown the self-organized criticality behavior of the valley fever model by running it with different test cases. We also introduced the concept of parameter morphism and compared the lumped and base models within the experimental frame of formation of new sites in the valley fever model. More research needs to be done to pinpoint the threshold value of the computed moisture in order for the lumped version of the valley fever model to mimic the behavior of the base version of the valley fever model. This will be quite useful, as the lumped model offers potential speedup advantages over the corresponding base model.

Chapter 7
Activity and Distributed Simulation

This chapter extends the work done by Zhang [30] and proposes an activity-based partition algorithm for performing distributed simulation of the valley fever model. It then shows some results of applying this algorithm to the valley fever model and compares its performance with a method based on an algorithm which doesn't exploit the spatio-temporal heterogeneity in the activity present in the valley fever model.

Recent research [24, 20, 30, 14] has shown that a DEVS based distributed simulation offers potential advantages over other methodologies which don't exploit spatio-temporal heterogeneity in the model. The valley fever model has spatial heterogeneity as it contains a ground model which has varying degree of favorableness. The valley fever model also has temporal heterogeneity as it contains models (Wind, Temperature, Moisture) which vary in time. A DEVS based simulation efficiently exploits this spatio-temporal heterogeneity in activity by allocating its computational resources to regions of high activity. Because of its spatio-temporal heterogeneity, the valley fever is an ideal model for using a DEVS based distributed simulation.

This chapter describes an algorithm to partition the ground model based upon its activity i.e. it describes an algorithm which allocates more computing nodes to patches which have a high degree of favorableness and less computing nodes to patches which have a low degree of favorableness. This algorithm is similar to the one given in [30]. The activity-based partition algorithm described in [30] used the state transition count for each cell and put the cells with high state transition count in the high-activity list and the cells with low state transition count in the low-activity list.

7.1 Activity Based Partition Algorithm

Figure 7.1 shows the activity-based partition algorithm that is used to partition the cells of the valley fever model according to their favorableness.

The algorithm whose flow chart is shown in figure 7.1 works as follows. For each cell in the cellspace of the valley fever model, it records the favorableness of the cell for the growth of the valley fever fungus. If the favorableness of a cell is greater than or equal to a certain threshold value(which in this particular case is 0.5), it adds that cell to the high-activity list as it has a higher chance of undergoing many state transitions due to its above-average favorableness. On the other hand, if the favorableness of a cell is less than the threshold value, the algorithm adds that cell to the low-activity list as it has a lower chance of undergoing state transitions due to its below-average favorableness.

7.2 Test Environment

The test environment that was used for testing the activity-based partition algorithm consists of a 5-node cluster where each node is composed of an AMD Athlon processor (2.0 GHz and 512 MB RAM). The DEVS-RMI [30] software was used for running these simulations. For the purpose of comparison, two different configurations are considered. The first one is a static partition i.e. this method of partitioning the cell space does not exploit the spatial heterogeneity in the activity. The second one is based on the proposed activity-based partition algorithm. In the static partition method, all the cells in the grid were evenly distributed among the available computing nodes. On the other hand, in the activity-based partition method, all the cells in the low-activity list were assigned to a single node and all the cells in the high-activity list were evenly distributed among the rest of the computing nodes. For simulating the variable favorableness of the ground, a random number was generated for each cell in the grid and if that random number was greater than a pre-determined threshold, then

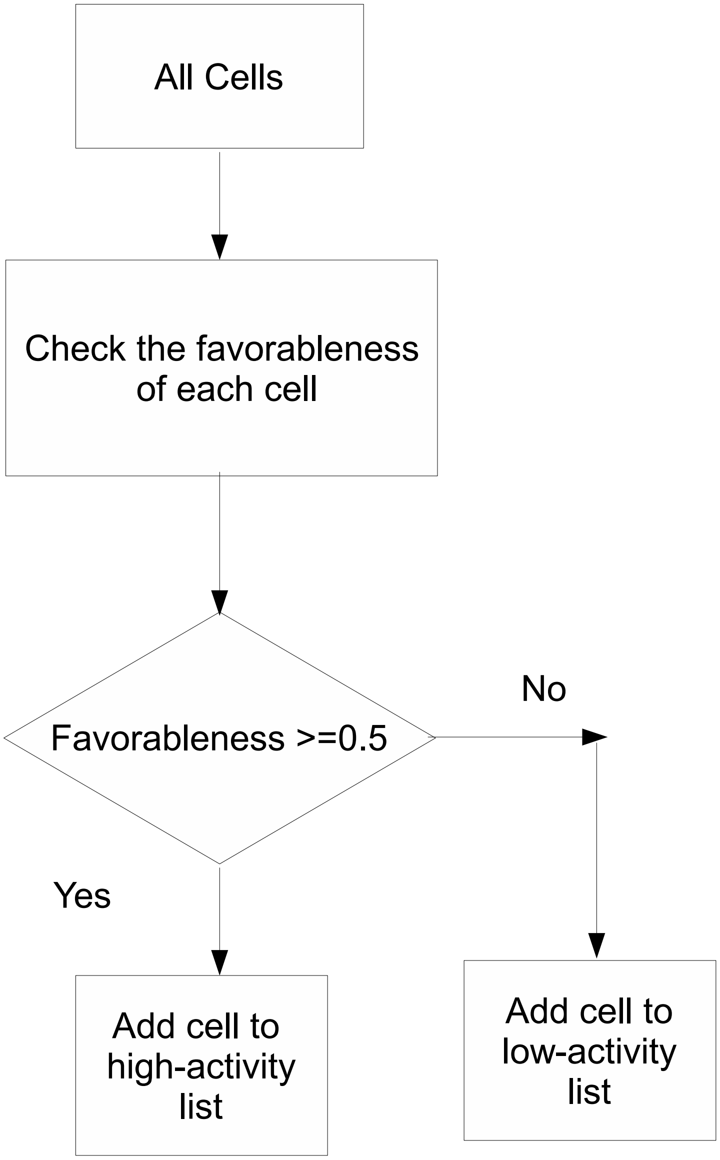

All Cells
Check the favorableness of each cell
Favorableness >=0.5
No
Yes
Add cell to high-activity list
Add cell to low-activity list

FIGURE 7.1. Activity-based partition algorithm.

Table 7.1. Run times for the static and activity-based partition algorithms when 3 computing nodes were used.

Grid Size	Static Partition	Activity Based Partition
3 × 3	14.07 secs 14.07 secs 14.07 secs	(threshold=0.5) 13.84 secs (threshold=0.8) 13.15 secs (threshold=0.95) 11.75 secs
5×5	35.0 secs 35.0 secs 35.0 secs	(threshold=0.5) 59.59 secs (threshold=0.8) 54.10 secs (threshold=0.95) 47.65 secs
7 × 7	145.0 secs 145.0 secs 145.0 secs	(threshold=0.5) 175.48 secs (threshold=0.8) 161.81 secs (threshold=0.95) 147.89 secs
8 × 8	201.68 secs 201.68 secs 201.68 secs	(threshold=0.5) 276.48 secs (threshold=0.8) 253.34 secs (threshold=0.95) 237.37 secs
10 × 10	551.74 secs 551.74 secs 551.74 secs	(threshold=0.5) 591.85 secs (threshold=0.8) 566.0 secs (threshold=0.95) 539.82 secs

the cell was added to the high-activity list; otherwise it was added to the low-activity list. Table 7.1 shows the result of applying the activity-based partition algorithm to different grid sizes when 3 computing nodes with 1 head node were used and the model was run for a simulation time of 10 days..

It can be noticed from table 7.1 that the performance of the activity-based partition algorithm increases as the grid size is increased. Also notice that as the threshold value is increased, the run time of the activity-based partition algorithm decreases significantly.

Table 7.2 shows the result of applying the activity-based partition algorithm to different grid sizes when 5 computing nodes with 1 head node were used and the model was run for a simulation time of 10 days. A threshold value of 0.95 was used for the activity-based partition algorithm.

It can be noticed from table 7.2 that the activity-based partition algorithm based method outperforms the static partition method.

Table 7.2. Run times for the static and activity-based partition algorithms when 5 computing nodes were used.

Grid Size	Static Partition	Activity Based Partition	Performance Gain
10 × 10	605.52 secs	536.69 secs	12.8%
15 × 15	2784.78 secs	2497.27 secs	11.5%

One of the main reasons for the low performance of an algorithm which uses the DEVS-RMI environment is the overhead due to communication between the nodes. Zhang [30] discusses techniques such as workload injection to improve the performance of the DEVS-RMI environment. More research needs to be done to explore techniques which can reduce the communication overhead in the DEVS-RMI environment.

Chapter 8
Conclusions and Future Work

In order to study spatio-temporal systems with heterogeneous activity we need to address two major problems associated with them. The first problem that needs to be addressed is that such complex real world systems extend over very large spatial and temporal domains and consume so many computational resources to simulate that it is not possible to study them with currently available computational resources. The second problem that needs to be addressed is that the data available for understanding such systems is limited because they are spread over space and time making it hard to obtain measurements both at the micro and macro levels. The second problem also makes it difficult to get the data for validation of their constituent processes while simultaneously considering their global behavior. One particular example of a spatio-temporal model with heterogeneous activity behavior is the valley fever fungus model which was considered in this dissertation. The valley fever fungus model was spread over a large spatial grid in the arid Southwest and it typically needed to be simulated over several decades of time to obtain useful information. It was also hard to get the temperature and moisture data (which are two critical factors on which the survival of the valley fever fungus depends) at every grid point of the spatial domain over the region of study. The valley fever model has spatial heterogeneity as the ground on which it is modeled has variable favorableness. It has temporal heterogeneity as the temperature and moisture models which decide the two critical factors on which the survival of the valley fever fungus depends vary with time. In order to address the first problem, we developed a method based on the discrete event system specification which exploits the heterogeneity in the activity of the spatio-temporal system by allocating more computational resources to the regions of high activity and less computational resources to the regions of low activity. Also, a method based on the discrete event system specification has been shown to be effective in solving relatively simple partial differential equation systems which are typical examples of systems which have heterogeneous activity behavior in space and time. The benefit of addressing the first problem was that it helped us to address the second problem.

In order to address the second problem we made use of a multilevel methodology which was based on modeling and simulation and systems theory. This methodology helped us in the construction of models with different resolutions(base and lumped models). This allowed us to refine an initially constructed lumped model with detailed physics-based temperature and moisture models and assess whether they improve on the original lumped models. For that assessment, we used the concept of experimental frame to delimit where the improvement was needed. This allowed us to work with the available data, improve the component models in their own experimental frame and then move them to the overall frame. In this dissertation, we developed a multilevel methodology and applied it to a valley fever model. Moreover, we studied the model's behavior in a particular experimental frame of interest, namely the formation of new sporing sites.

In this dissertation, we also have shown a simple activity-based partition algorithm for doing distributed simulation of the valley fever model. The experiments suggest that the performance advantage of using the activity-based partition algorithm increases as the cell space is increased. More research needs to be done in this area to develop techniques to reduce the communication overhead caused due to DEVS-RMI.

We have experimented with the concept of parameter morphism to see if we can get the lumped model to behave as close as possible to the base model within a given experimental frame. More research needs to be done in this area to develop a mathematical framework to modify the parameters which bring the behavior of the lumped model as close as possible to the base model within a given experimental frame.

References

[1] Per Bak. *Self-Organized Criticality: Emergent Complex Behavior in Physical and Biological Systems.* Henrik J Jensen, Jan 1998.

[2] Per Bak. *How Nature Works: The Science of Self-Organized Criticality.* Springer, May 1999.

[3] Philip R. Bevington. *Data Reduction and Error Analysis for the Physical Sciences.* McGraw-Hill, 2003.

[4] H.S. Carslaw and Jaeger. *Conduction of heat in solids.* Oxford University Press, USA, 1986.

[5] J. Crank. *The Mathematics of Diffusion.* Oxford University Press, 1975.

[6] Tino Duong. Self-organized criticality.

[7] Robert Evans, D. K. Cassel, and R.E. Sneed. Calibrating soil-water measuring devices, June 1996.

[8] M. E. Gettings. Unpublished data. USGS.

[9] Mark E. Gettings and Mark W. Bultman. Quantifying favorableness for occurrence of a mineral deposit type using fuzzy logic - an example from arizona. Technical report, U.S. Geological Survey, 1993.

[10] M.E. Gettings and F.S. Fisher. Agent-based modeling of physical factors that may control the growth of coccidioides (valley fever fungus) in soils. *Submitted to the G^3*, 2008.

[11] R. Jammalamadaka. Activity characterization of spatial models: Application to the discrete event solution of partial differential equations. Master's thesis, University of Arizona, Tucson, Arizona, USA, 2003.

[12] R. Jammalamadaka, M. Gettings, M. Bultman, B. P. Zeigler, and M. Zhang. Complex system simulation: Devs implementation of the valley fever model. In *2007 Spring Simulation Multiconference*, Norfolk, Virginia, USA, 2007. SCS.

[13] R. Jammalamadaka, J. Nutaro, M. Gettings, and B. P. Zeigler. Devs reimplementation of an agent-based valley fever model. In *2005 Spring Simulation Multiconference*, San Diego, 2005. SCS.

[14] Rajanikanth Jammalamadaka and Bernard P. Zeigler. A generic pattern for modifying traditional pde solvers to exploit heterogeneity in asynchronous behavior. In *PADS*, pages 45–52, 2007.

[15] Ernesto Kofman. Discrete event simulation of hybrid systems. *SIAM Journal on Scientific Computing*, 25(5):1771–1797, 2004.

[16] Alexandre Muzy, A. Aiello, Paul-Antoine Santoni, Bernard P. Zeigler, James J. Nutaro, and Rajanikanth Jammalamadaka. Discrete event simulation of large-scale spatial continuous systems. In *International Conference on Systems, Man and Cybernetics (SMC)*, Hawaii, USA, October 2005. IEEE.

[17] Alexandre Muzy, Eric Innocenti, Antoine Aiello, Jean François Santucci, Paul-Antoine Santoni, and David Hill. Dynamic structure cellular automata in a fire spreading application. In Helder Araújo, Alves Vieira, José Braz, Bruno Encarnação, and Marina Carvalho, editors, *ICINCO (3)*, pages 143–151. INSTICC Press, 2004.

[18] Alexandre Muzy, Eric Innocenti, Jean François Santucci, and David R. C. Hil. Optimization of cell spaces simulation for the modeling of fire spreading. In *Proceedings of the 36th Annual Simulation Symposium*, pages 289–296, Orlando, FL, March 2003. IEEE Computer Society.

[19] D.L. Nofziger. Diffusion of solute in soil media. Technical report, Oklahama State University, 2003.

[20] James Nutaro. *Parallel Discrete Event Simulation with Application to Continuous Systems*. PhD thesis, University of Arizona, Tuscon, Arizona, 2003.

[21] James Nutaro. Constructing multi-point discrete event integration schemes. In *Proceedings of the 2005 Winter Simulation Conference*, 2005.

[22] Jim Nutaro. A discrete event system simulator.

[23] D. Pappagianis. Epidemiology of coccidioidomycosis. *Current topics in Medical Mycology*, 2:199–238, 1988.

[24] Sunwoo Park. *Cost-based partitioning for distributed simulation of hierarchical modular DEVS models*. PhD thesis, University of Arizona, 2003.

[25] Scott N. Putnam and David S. Chapman. A geothermal climate change observatory: First year results from emigrant pass in northwest utah. *Journal of Geophysical Research*, 101:21,877–21,890, October 1996.

[26] U. Wilensky. *NetLogo,*. Center for Connected Learning and Computer-Based Modeling, Northwestern University., Evanston, IL, 1999.

[27] Bernard P. Zeigler. *Theory of Modeling and Simulation.* John Wiley and Sons, 1976.

[28] Bernard P. Zeigler. *Theoretical Systems Ecology: Advances and Case Studies.* Academic Press, New York, 1979.

[29] Bernard P. Zeigler, Herbert Praehofer, and Tag Gon Kim. *Theory of Modeling and Simulation, 2nd Edition.* Academic Press, San Diego, CA, 2000.

[30] Ming Zhang. *Toward a flexible and reconfigurable distributed simulation: A new approach to distributed DEVS.* PhD thesis, University of Arizona, 2007.