SCALABLE COMPUTING: PRACTICE AND EXPERIENCE
Volume 8, Number 2, pp. 185–196. http://www.scpe.org
ISSN 1895-1767
© 2007 SWPS

# AGENT-BASED MODELING AND SIMULATION OF BIOMOLECULAR REACTIONS

VAISHALI VALLURUPALLI AND CARLA PURDY*

**Abstract.** We present an agent-based, three-dimensional model of phage lambda, a virus infecting *E. coli* which exhibits two phases, lysogenic and lytic. This process is useful for bio-based sensors or switches and is a widely studied gene regulatory system. We model system constituents as software agents. Complex system behavior emerges from local agent interactions. Agent-based modeling lets us study how individual parameters affect overall system behavior. This bottom-up approach is an alternative to top-down approaches using differential equations and stochastic simulation. It can model any system of biomolecular reactions, with applications in physiology, pharmacology, medicine, environmental monitoring, and homeland security.

**Key words.** agent-based modeling, ABM, complex systems, systems biology, phage lambda

**1. Introduction.** Traditionally, biomolecular reactions have been studied and analyzed using ordinary differential equations (ODE) or stochastic simulation. The ODE model has been employed extensively for modeling systems with either a few or a moderate number of factors and in which effects of interest tend to be sparse and linear or of low-order. In these systems, higher-order interactions are typically negligible and errors are normally distributed [20]. Noise is typically aggregated into one term [4]. Stochastic models are effective for systems with a large number of components, with each component having simple interactions and few states. However, certain systems, such as biological cells, have a significant number of components whose interactions are complex [3]. The main objective of this work is to model and simulate a gene regulatory system and its constituent reactions using agent-based modeling (ABM) and to compare this approach against the traditional approaches. We have modeled our system using the Unified Modeling Language (UML) and simulated it using a 3D visualization engine, BREVE [12].

**1.1. Agent-based Modeling (ABM).** Most biological systems are complex in nature. A complex system has many of its components coupled in a non-linear fashion. The variables in a complex system can exhibit complicated, discontinuous behaviors over time. Most complex systems exhibit the emergence property, i. e., the formation of complex patterns from simpler interaction rules. Thus the global behavior of the system can be determined by defining the lower-level interaction rules among the components [9]. One such complex biological system is the gene regulatory system of the phage lambda virus injected into an *E. coli* cell, which we model here.

Developing software for agent-based systems can make use of many modern software engineering techniques, including decomposition, or dividing a problem into small, manageable parts, abstraction, or choosing which details of a problem to model and which to suppress, and organization, or identifying and managing the relationships among the various system components [5, 24]. The agent-based model is a bottom-up paradigm wherein the lowest level entities, called agents, interact with each other autonomously.

An *agent* is an encapsulated computer system that is situated in some environment and that is capable of flexible, autonomous action in that environment in order to meet its design objectives [24]. The idea is to construct the model using agents and to simulate the interactions of those agents in parallel to model the real phenomena on a system level. The agents are situated in space and time and have some properties and certain sets of local interaction rules. Though intelligent, they cannot by themselves deduce the global behavior resulting from their dynamic interactions. The system evolves from the micro level to the macro level. Thus agent-based modeling uses a bottom-up design strategy rather than a top-down strategy. Agents are commonly assumed to have well-defined bounds and interfaces, as well as spatial and temporal properties, including such dynamic properties as movement, velocity, acceleration, and collision. They exist in an environment which they sense and can communicate with through their interfaces. They are assumed to respond in a timely manner to changes in their environment. They are autonomous, encapsulating a state and changing state based on their current state and information they receive [4, 24].

In agent-based modeling the problem is decomposed so that agents are equipped with knowledge to solve the problem. As a result, the control complexity is reduced due to the decentralization achieved by decomposition. An agent-based approach helps us to study the emergence of complexity of the system, in that the scale and effect of various parts on the global behavior and vice-versa can be studied in greater detail and with much more accuracy. Thus, in contrast to systems based on differential equations, an agent-based system may possess many factors, may exhibit non-linear and even non-polynomial behavior, may have many higher-order interactions, may have a variety of possible errors, with differing distributions, and may provide many different types of performance measures [20].

*University of Cincinnati, Electrical and Computer Engineering Department, Cincinnati, OH 45221,{vallurv, carla.purdy}@uc.edu

One important benefit of agent-based systems is that they can exhibit emergent behavior, or higher-level behavior which is a consequence of agent interactions. For an ABM, it is often said that "the whole is more than the sum of its parts" [4]. Agent-based systems also allow for easy modification of interaction rules or behavior, as well as for viewing agents or groups of agents at different levels of abstraction, as appropriate. Agent-based systems also allow for the study of system fluctuations, especially those in which a small change in a variable or a discontinuity can have extreme consequences on the system as a whole [4].

**1.2. Comparison of ODE and Stochastic Simulation Approaches to the Agent-based Model.** Modeling the phage lambda gene regulatory system using a differential equations approach as in [13, 23] requires that the various processes be converted into chemical equations which are later converted into differential equations. One obvious limitation is that the ODE method does not account for spatial aspects. Dimerization of monomer proteins, dissociation of dimers, Repressor binding to Operator regions, and RNA Polymerase binding to Promoter regions are all processes in which the spatial aspects of the DNA play a crucial role. These details are not modeled in the ODE approach. Agent-based models account for these spatial and temporal aspects and hence can be closer to actual biological processes. Another simplification in ODEs is that the protein molecules are modeled as concentrations and the actual reactions as change of these concentrations with time. This leads to modeling the system in a structured and programmed manner where reactions happen in sequence (as in the ODE approach) at a given rate. However, actual biological processes happen concurrently and the rates at which these processes proceed may be affected by many factors, not all of which are modeled by the ODE approach.

Stochastic simulations also do not usually take into account the spatial aspects of the simulation. In many stochastic simulations, reactions also proceed sequentially, although in a random order. In addition, stochastic simulations may not incorporate geometric information. With an agent-based approach, many processes work concurrently and hence complex and often unpredictable behaviors can be observed.

**1.3. Issues with ABM.** Modeling a system with a bottom-up approach requires that every individual agent's behavior be described. The greater the number of details that go into describing the behavior of the system, the greater is the computational power that is required to simulate the behaviors of all constituent agents. This is a limitation in modeling large systems using ABM [4]. A reasonable approach is to provide several levels of abstraction and granularity, which can be chosen depending on the level of detail needed and the computational resources available. Here we are modeling reactions at the molecular level, for a relatively small number of molecules, so we have chosen a very fine-grained level of modeling.

The objective of simulating a system is usually to predict the outputs at untried inputs or to optimize a function of the input parameters. ABM models cannot do either. Rather, a more relevant analysis using ABM is to study the system, search for insights, and get a basic understanding of the agent-based model. The insights that we can hope to get are in identifying the important factors and their interactions, as well as their effects, and the locations, thresholds, and ranges where interesting things like fluctuations can occur [20]. ABM models can also be combined with heuristic techniques which search for parameter values to optimize a desired behavior [17].

**1.4. Unified Modeling Language (UML) and ABM.** The Unified Modeling Language (UML) is a widely-used graphical language for describing overall system design. It includes structures for describing individual classes (class diagram), related classes (ER or entity-relationship diagram), and system and object states [5, 6]. Dinsoreanu et al [8] have presented a methodology for designing agent-based systems and models expressed in UML notation. The methodology has the following steps:

1. Specification of functionality: the functionality of the system is expressed in terms of the various tasks of the agents.
2. Design of the organizational model: agents constituting the system, their respective tasks, and rules that need to be followed by each of them are identified.
3. Definition of the interaction model: the communication protocols between the agents are defined.
4. Design of the environment model: the environment in which the agents operate and interact is defined.

Webb and White [22] have modeled and simulated the glycolic pathway within the cytoplasm and the TCA cycle that takes place within the mitochondrial matrix using the CellAK approach. The system is decomposed into an inheritance hierarchy of classes and a containment hierarchy of actors using UML class diagrams. Each actor, also called a capsule, possesses a state diagram which models the reactivity of the actors to real-time events and internally-generated timeouts. The model was validated against Gepasi [14].

Cannata et al [7] proposed the Multi Agent System (MAS) approach to effectively model a biological system using mediating artifacts which enable agents to engage in different types of interactions. The approach has the advantage that

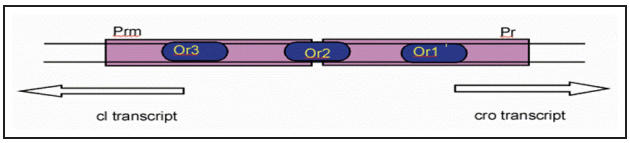


FIG. 2.1. *Basic structure of the phage lambda DNA.*

it can provide the right level of abstraction to model a biological system. For simulating biological systems, they propose models which allow us to analyze the system from different viewpoints, i. e.,

- static-structural view with a biomolecular knowledge of components, their properties and their relationships
- dynamic view which shows how the components react to the environment and other agents over a period of time
- functional view which shows how the functions are performed by the different agents

**1.5. BREVE.** Many simulators were studied for ease of use and availability for our purpose of simulating decentralized behaviors. There are a few popular packages for simulation of decentralized systems which provide 2D visualization. Swarm [16] and StarLogo [19] are some of the popular packages commonly used for agent behaviors but, compared to BREVE, they do not provide 3D simulations or visualizations. On the other hand, there are some commercial simulators like the one described in [2] which provide 3D graphics and physical simulations but which are quite costly.

BREVE [12] is a 3D simulation engine developed to provide quick and easy simulations of decentralized systems. It is an open-source package available for Mac OS, Linux, and Windows platforms. It allows users to define and visualize decentralized agent behaviors in continuous time and continuous 3D space. The package comes with its own interpreted object-oriented scripting language called steve, an OpenGL display engine, and collision detection. BREVE also has a library of built-in classes and provides graphical 3D visualization and collisions between 3D bodies. These features remove a significant amount of programming overhead. A special emphasis is placed on simulating lower-level interactions in order to evolve highly realistic macroscopic behaviors.

**2. System to Be Studied.** *Escherichia coli (E. coli)* is an intestinal bacterium and is a commonly studied prototype for bacteria. When a passively infected *E. coli* bacterium is exposed to a dose of ultraviolet light, it stops reproducing and starts to produce a crop of viruses called phage lambda into the culture medium. This process of rapid reproduction of the phage lambda viruses is called lysis. The newly-formed lambda phages multiply by infecting fresh bacteria. Some of the infected bacteria lyse further, causing more phages to be produced, whereas some bacteria may carry the phage in a passive form, causing the bacteria to grow and divide normally. This process of passive reproduction of the phage lambda is called lysogeny.

This switching of the virus between two states, i. e., from the passive form or lysogeny to the activated form or lysis is similar to the ON and OFF states of a digital inverter. Jacob and Monod [10] showed that this switching is a basic example for the turning on and off of genes. Thus an accurate model of the lysis and lysogeny processes in *E. coli* is of interest in itself and can also form the basis for modeling more complex processes in other organisms. The two states are described more completely by:

*Lysogenic state*: Only a single phage chromosome is turned on. It grows and replicates passively, as part of the *E.coli* bacterium. However, when irradiated with ultraviolet light, almost every lysogen in the bacterium enters the lytic state.

*Lytic state*: Various sets of phage genes turn on and off in a regulated manner. Consequently, the lambda chromosome is extensively reproduced and 45 minutes later the bacterium lyses and releases new phage [18].

**2.1. Gene Expression.** Genes which are expressed are said to be on, whereas those not expressed are said to be off. The expression of the genes is regulated. The regulation of genes can be easily visualized in the switching of the phage lambda from the lysogenic to the lytic state.

Transcription is the first step in gene expression. The sequence of base pairs along one of the gene strands is copied into a linear molecule called RNA. A gene is on, or being expressed, if it is being copied into RNA and off if it is not. There are various kinds of RNA molecules, some of which are end products, while others are messenger RNAs (mRNAs) which specify the designs of proteins. Genes are transcribed into mRNA by an enzyme called RNA Polymerase.

The process begins with the RNA Polymerase binding to a site on the gene called Promoter. The Promoter is a region of the gene extending over 60 base pairs. We consider two Promoters, Pr and Prm, in our model. After binding, RNA Polymerase travels away from the Promoter region along the gene, synthesizing mRNA as it moves, resulting in unwinding regions of DNA. This unwound sequence then forms the template for the complementary mRNA strand. Figure 2.1 shows a segment of DNA with the directions of the two Promoters indicated.

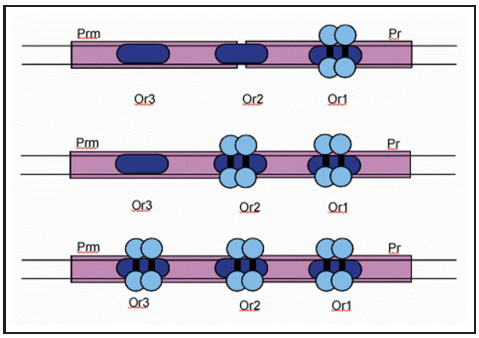


FIG. 2.2. *Repressor action.*

RNA Polymerase can be helped or prevented in its process of transcription of the gene by the regulatory proteins that bind to specific sites on the DNA called Operators. In Figure 2.1, three Operator sites, Or1, Or2, and Or3, are shown. A negative regulator protein prevents transcription, whereas a positive regulator protein aids transcription. Repressor or Rp is a protein of 236 amino acids which fold into two distinct domains called the amino and carboxyl domains. Two Repressor monomers can associate to form a Repressor dimer. The Repressor dimers use their amino domains to bind to DNA at an Operator site [18].

Cro is a protein made up of 66 amino acids folded into a single domain. Cro monomers have high affinity for each other and hence form a dimer. A Cro dimer binds to the Operator sites on the DNA.

Operators are specific sites on the DNA molecule which may vary in the strength with which they bind the protein. There are three Operator sites, OR1, OR2 and OR3 which are relevant to our model of the lambda phage. Each of these Operator sites vary in their affinity for Rp and Cro. They are depicted in Figure 2.1 above.

**2.2. Lysogeny.** Repressor dimers have higher affinity for OR1 and hence bind to it first. If OR1 is filled, OR2's affinity is increased. Hence in the presence of high concentrations of the Repressor dimer, OR2 is filled almost simultaneously with OR1. Thus binding of Repressor to Operator sites is cooperative. When a Repressor binds to site OR2, it covers the part of the DNA which is necessary for the Polymerase to bind to PR. Thus, it prevents RNA Polymerase from binding to the Cro gene and turns it off. Consequently, it helps Polymerase to bind and begin transcription of the cl gene in a lysogen. Thus, Repressor exerts both positive and negative control. This is shown in Figure 2.2.

**2.3. Lysis.** Repressor at both OR1 and OR2 keeps Cro off and stimulates transcription of its own gene, cl. Repressor is synthesized continuously as long as cells grow and divide. If Repressor concentration increases, it binds to OR3 and turns its own gene off, thus preventing further synthesis of Repressor. Once the Repressor concentration falls down to a level such that OR3 is empty, the cl gene is turned on and normal synthesis continues. Thus a constant Repressor concentration is maintained, despite fluctuations in the growth rate.

To change the state of the switch, ultraviolet (UV) light is applied. UV light leads to a significant change in the behavior of a bacterial protein called RecA, causing it to cleave lambda Repressor monomers. The cleavage effectively inactivates Repressors since they are no longer able to dimerize and the monomeric forms have low affinity for the Operator. As a result, there are few dimers to replace the ones which fall off the Operator sites. As Repressor dimers fall off OR1 and OR2, the synthesis of Repressor is stopped. Hence no further Repressor to bind to the empty Operator sites is available. Consequently, Polymerase binds to PR to begin transcription of the Cro gene.

Cro dimers bind non-cooperatively to the Operator sites. The affinity of Cro for the three Operator sites is opposite to that of the Repressor. Hence, if a Cro dimer approaches the Operator sites, all of which are empty, then Cro binds to OR3. This prevents binding of Polymerase to PRM and abolishes further synthesis of Repressor. At this point, the switch is said to be flipped and lytic growth continues. As PR is turned on and Cro is transcribed, more Cro is synthesized until it fills OR1 and OR2. In summary, Cro first stops Repressor synthesis and after some time turns off its own gene. This is depicted in Figure 2.3.

The phage lambda virus has one of the simplest gene regulatory systems which shows an interesting phenomenon analogous to a switch (from lysogenic to lytic state and vice versa), when injected into *E. coli* bacteria. The various processes that go into the gene transcription, such as Repressor dimerization, dissociation and decay, and translation, are modeled in our system, along with spatial aspects like the Promoter and Operator regions of the DNA. The detailed biological mechanisms and nomenclature can be found in [18].

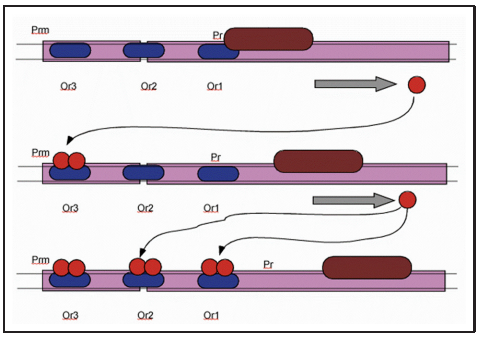


FIG. 2.3. *Cro action.*

Tsui has developed an agent-based model of the lambda switch. Her paper shows that agent-based modeling is an effective approach for studying gene regulatory systems at the molecular level. Our work is an extension of her work [21]. Much work has been done on phage lambda using the differential equation and stochastic models. See, for example, [1, 13, 15, 23]. But much more accurate models can be developed. We would like to study such factors as space, concentration of proteins, and the molecular-level dynamics which influence the lytic and lysogenic processes of a phage lambda system and compare the qualitative and quantitative results against those predicted by models based on ordinary differential equations or stochastic simulation.

**3. Modeling Specifics.** We have modeled a basic phage lambda DNA structure and protein behaviors based on Ptashne's extensive work on the phage lambda virus and have followed his notations of names and structure [18]. The reactions shown in Figure 3.1 [23] have been modeled using the agent-based approach.

**3.1. Agent Rules.** BREVE, the language we will use, provides for in-built classes of the type Mobile and Stationary, both derived from a base class, Object. Agents of type Repressor, Cro, RecA, and RNA Polymerase (RNA Poly) are modeled as agents of type Mobile. Each of these agents has an identification, shape, color, velocity and lifetime. Some of the local agent rules are:

- Repressor, Cro, RecA, and RNA Poly are initialized as monomers
- All mobile molecules can move in a fixed volume around the DNA
- All mobile molecules have random motion with a fixed, equal velocity
- All molecule agents have a lifetime for which they would be present in the environment
- A monomer (Repressor or Cro) will form a dimer upon collision with another monomer
- A dimer (Repressor or Cro) will dissociate into its constituent monomers after a certain period of time within its lifetime
- Agents decay after their lifetime has ended
- RecA on colliding with a Repressor monomer causes it to be incapable of forming a dimer
- Dimer agents (Repressor and Cro) sense the affinity and distance to Operator sites. If any of the Operator sites are not filled, they attach to the appropriate site. Else, they continue to move randomly
- Promoter agents sense the status of the Operator sites and accordingly set their own states
- RNA Polymerase agent senses the status of the Promoters. If one of them is turned on, then it binds to it and starts synthesizing either Repressor or Cro until it reaches the end of the gene sequence

The Controller class, which is analogous to the main() class in the C++ language, is responsible for initializing all the agents, monitoring their behavior, and updating the global behavior of the system. It does not, however, centrally control the system. Thus local agent behaviors are allowed to evolve over time.

**3.2. UML Description.** For our system we have used the UML constructs class, entity-relationship diagram, and state diagram.

**3.2.1. Class and Entity-Relationship Diagrams.** *Class* is a template for a set of objects that share a common set of attributes and operations. Figure 3.2 gives an example of a class. The first section specifies the name of the class. The second section specifies the attributes of the class. For example, type is an attribute of the class RpMonomer. The third section specifies the operations performed by the class. For example, to get-type is an operation performed by the RpMonomer class to access the type attribute. In our system, each class corresponds to a type of agent.

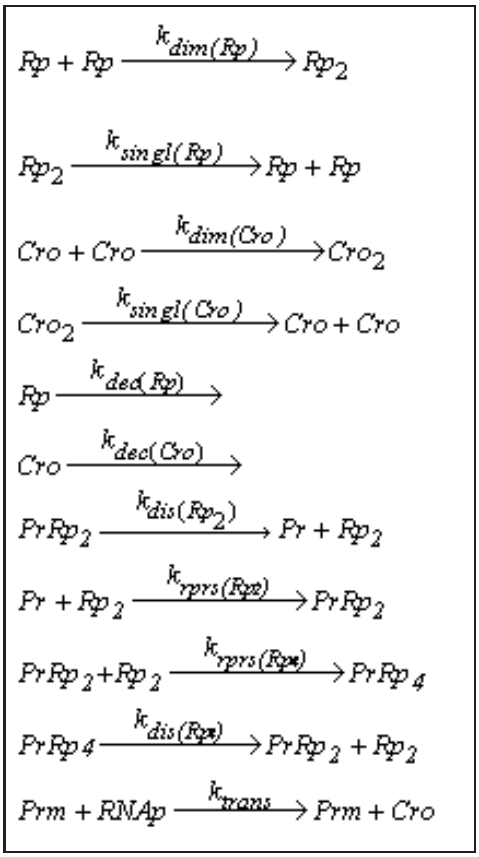


FIG. 3.1. *The reactions comprising the phage lambda switch.*

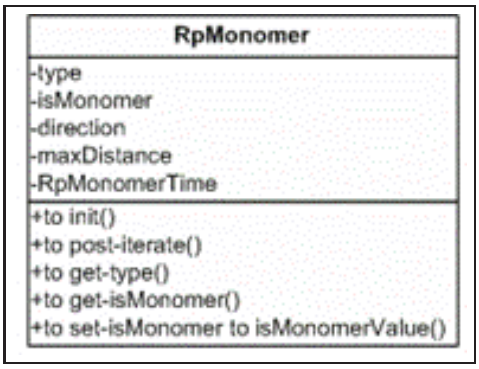


FIG. 3.2. *Example class–RpMonomer.*

*Entity-relationship* or *ER diagram* shows the relationships and interfaces between classes. In Figure 3.3, the legends for the three basic types of relationships are given. Figures 3.4 and 3.5 show the ER diagrams for the system's stationary and mobile objects, respectively. *Is-a relationship* between classes indicates inheritance of attributes and operations by the class which is lower in the hierarchy. In Figure 3.4, Operator and Promoter are classes which both inherit the properties of the class DNARegion. *Has-a relationship* indicates that the class includes components from another class. In Figure 3.5, RpMonomer is a class which is a component of Class RpDimer. All other relationships in the ER diagrams are of the generic type *uses*.

**3.2.2. System State Diagrams.** The agent-based approach allows us to view the state of the system from multiple points of view. The phage lambda system after initialization with enough Repressor concentration goes into the lysogenic state. It continues to be in the lysogenic state until UV light is applied. It then goes into the lytic state. This is shown in

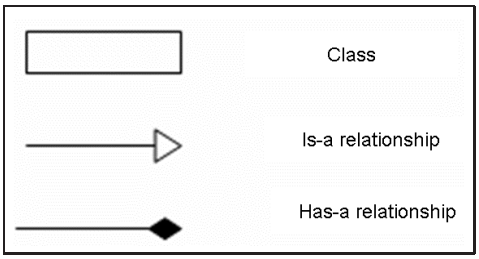


FIG. 3.3. *Legend for class diagram notations.*

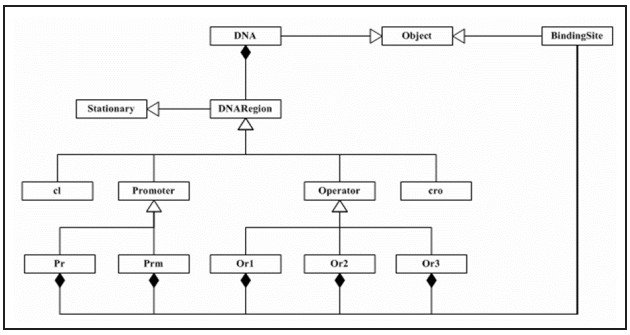


FIG. 3.4. *ER diagram for system's stationary objects.*

Figure 3.6. The Rp and Cro dimer agents, after being initialized, are in a state of random motion. However, if they sense that one or more Operator sites are empty, then they bind to particular site(s). This is shown in Figure 3.7.

**4. Results.** We have modeled the phage lambda system as defined in UML. An example snapshot of the simulation is shown in Figure 4.1. In this figure, the light purple region represents the Promoter Pr (right) and the light brown region is Prm (left). The dark blue regions are the Operators, Or1, Or2, and Or3 (right to left). Small red spheres are Rp monomer agents, whereas large red spheres are Rp dimer agents. The blue spheres represent Cro, the larger ones being the Cro dimer and the smaller ones being the Cro monomer agents. The black spheres are the RecA agents and the grey spheres represent the cleaved Repressor monomers.

**4.1. BREVE Simulation Results.** Some details of our program, written in BREVE's steve language, are given here.

- RPMONOMER_COUNT and RECA_COUNT represent the intial number of molecules of Repressor and RecA monomers, respectively, used in the simulation
- RECA_INIT_TIME is the time after start of simulation that RecA monomers are initialized (applying UV light)
- X_LIFETIME is the total time for which a molecule agent of type X is active in the simulation
- X_VEL is the velocity for a molecule agent of type X
- RP_TO_OR1_TIME, RP_TO_OR2_TIME, RP_TO_OR3_TIME are the times for which the Repressor is bound to the Operator sites
- CRO_TO_OR1_TIME, CRO_TO_OR2_TIME, CRO_TO_OR3_TIME are the times for which the Cro agents are bound to the Operator sites

Table 4.1 gives the values of the parameters used in the basic simulation. The graph for these values is shown in Figure 4.2. The initial concentration of Repressor monomers is 20. As the simulation proceeds, the concentration of Repressor falls down gradually, especially after t = 15 seconds, when RecA is initialized. The concentration of Cro is found to increase gradually thereafter.

We also did a series of experiments to study the effects of changes in the various parameters. We present some examples here.

- Figure 4.3 shows the effect of increase in initial Repressor concentration. We increased it to 25. It can be seen that Cro was synthesized only after 20 seconds as against 17.5 seconds in Figure 4.2. The rate of Repressor decay is much slower than in the former case
- Figure 4.4 shows the effect of decreasing initial Repressor amount to 15 molecules. The effect of the change is the rapid synthesis of Cro

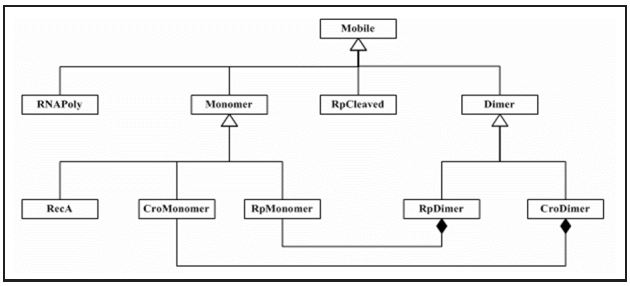


FIG. 3.5. *ER diagram for system's mobile objects.*

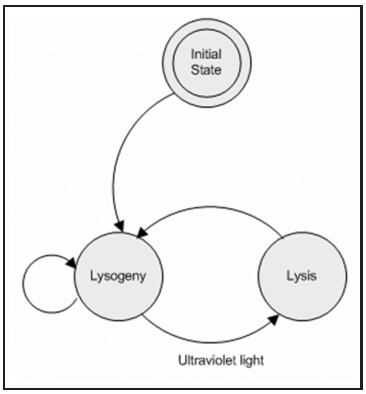


FIG. 3.6. *State machine representing the system states.*

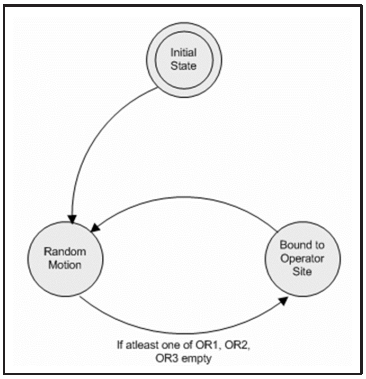


FIG. 3.7. *State machine representing the states of a Repressor*

- Figure 4.5 shows the effect of modifying the RecA initialization time to 10 seconds instead of 15 seconds. We observe that Cro starts being produced after just 12 seconds
- Figure 4.6 shows the effect of decreasing the lifetime of the Repressor dimer to 4 seconds. Decreasing the lifetime of a Repressor dimer effectively reduces the time it is bound to the Operator site Hence, the amounts of both Rp and Cro are prone to fluctuate
- Figure 4.7 shows the effect of increasing the times that the Repressor is bound to the Operator sites. As a result, the synthesis of Repressor takes place for a longer time than is the case in Figure 4.2. So, for the same amount of RecA molecules, with the same velocities, it takes a longer time to empty OR1 and OR2 and hence Cro is not synthesized at as fast a rate

**4.2. Comparison to ODE and Stochastic Simulation Results.** Krishnan and Purdyl [13] have simulated some of the phage lambda system equations of Figure 3.1. For an initial 1 micromolar Repressor concentration, the concentration of Cro in the absence of Rp was found to be 0.2 micromolar.

At the system level, the kinetics of a reaction are determined by the reaction rate constant and relative concentrations of the reactants. At the molecular level, however, the kinetics are a function of the probability of collisions between individual molecules of the participating reactants per unit time [11]. They are also influenced by the reactants' initial concentrations as well as their respective affinities. Khan et al [11] have estimated relative reaction times in terms of the relative rate constants to be

$$\frac{v1}{v2} = \frac{k1}{k2} \qquad or \qquad \frac{1/v2}{1/v1} = \frac{1/k2}{1/k1} \tag{4.1}$$

where v1 and v2 are the velocities and k1 and k2 are the kinetic constants of two reactions 1 and 2 which have equal initial concentrations of reactants. The reaction time thus obtained is scaled against the reaction which is the slowest, having the least rate constant among a set of reactions.

For our calculations, we have calculated the concentrations of each agent molecule using the formula

$$C = \frac{N/A}{V} \tag{4.2}$$

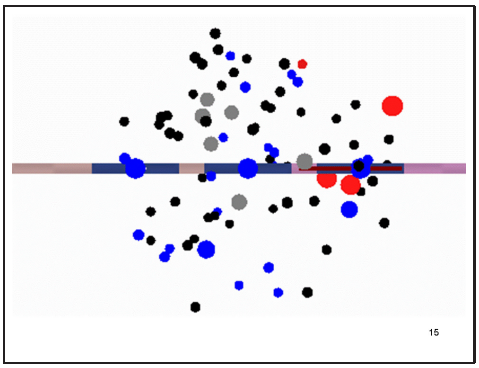


FIG. 4.1. *Cro and RecA in action.*

TABLE 4.1
*The base parameter values for simulation of the phage lambda system.*

| | |
|---|---|
| RPMONOMER_COUNT | 20 |
| RECA_COUNT | 50 |
| RECA_INIT_TIME | 15 |
| RPMONOMER_LIFETIME | 20 |
| CROMONOMER_LIFETIME | 20 |
| RECAMONOMER_LIFETIME | 60 |
| RP_TO_OR1_TIME | 1 |
| RP_TO_OR2_TIME | 2 |
| RP_TO_OR3_TIME | 3 |
| RPDIMER_LIFETIME | 7 |
| CRO_TO_OR1_TIME | 3 |
| CRO_TO_OR2_TIME | 2 |
| CRO_TO_OR3_TIME | 1 |
| CRODIMER_LIFETIME | 7 |
| RPDIMER_VEL | 20 |
| CRODIMER_VEL | 20 |
| RECA_VEL | 60 |
| RPMONOMER_VEL | 40 |
| CROMONOMER_VEL | 20 |
| RNA_VEL | 5 |

where C is the concentration of molecules of a given type, N is the number of molecules of that type, A is the Avogadro number (6.023 x $10^{23}$) of molecules in a mole and V is the volume within which these reactions occur. In our BREVE simulation, we have modeled the DNA structure in terms of the base pairs as obtained from [18]. As a result, the volume was found to be approximately 0.15 x $10^{-15}$ liters.

Krishnan and Purdy [13] have also simulated a stochastic model of the phage lambda system and have obtained the result shown in Figure 4.8.

In order to compare our results with those of Figure 4.8, we have scaled our system with the same input concentrations for Rp and RNA Polymerase [13]. The rate constants of Figure 3.1, as obtained from [18], were also modeled such that all reactions were scaled against the slowest reaction among the set of reactions. With the kinetic constants scaled as those in [23] and [13], and the same input concentrations as in [13], we initialize RecA agents after just 0.5 seconds of simulation time (Figure 4.9). For this simulation, we use the parameter values given in Table 4.2. We see that the general shapes of the curves from 0-50 seconds in Figures 4.8 and 4.9 seem to be similar. But there are several factors which differ in the two simulation models, and we are still experimenting with the correct values for each model. Thus we are not yet able to compare the two models in enough detail to determine definitively how well they match.

**5. Conclusions and Future Work.** In modeling biological processes, it is important to incorporate the spatial and temporal properties of the molecules and genes to provide for a more accurate model of the system. Hence, we modeled and simulated the gene expression of the phage lambda virus using the ABM paradigm. The behavior of the gene and the regulatory proteins has been modeled for three-dimensional visualization using BREVE.

Comparing our work to [13], we find that our agent-based approach has the advantage of visualizing and understanding the system at the molecular level. By including the knowledge of actual kinetics at the molecular level, the accuracy

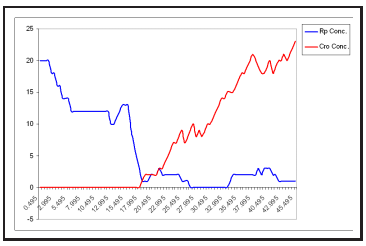


FIG. 4.2. *Rp vs. Cro, simulation with values of Table 4.1.*

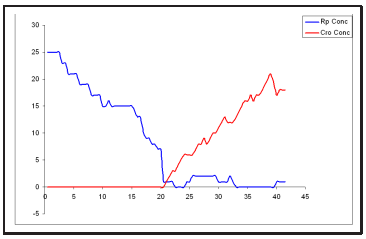


FIG. 4.3. *Rp vs. Cro, simulation with RpMONOMER_COUNT = 25.*

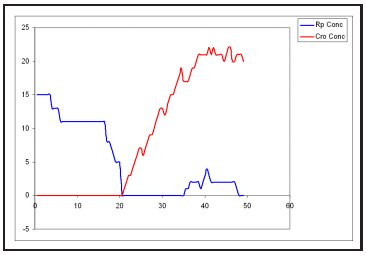


FIG. 4.4. *Rp vs. Cro, simulation with RpMONOMER_COUNT = 15.*

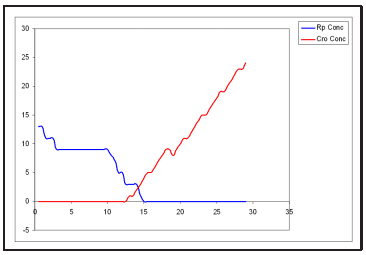


FIG. 4.5. *Rp vs. Cro, simulation with Rp_MONOMER_COUNT = 15 and RecA_INIT_TIME = 10 seconds.*

of the model can be improved significantly. As we can see from the graphs, we can observe the functioning of the system in the presence of random fluctuations in the expression of genes. Also, the effect of varying various parameters such as concentrations and velocities can be observed. Our model was found to be easy to scale in terms of both the reaction parameters and the level of detail. Hence, we anticipate that the phage lambda system can be modeled to include detailed descriptions of the gene structure and the behaviors of the molecular agents. The model could also be ported to a parallel environment for the simulation of more complex systems.

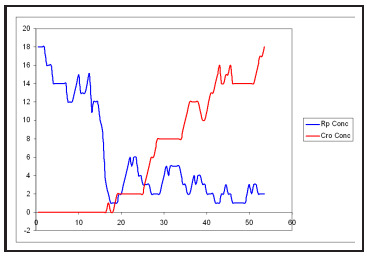


FIG. 4.6. *Rp vs. Cro, simulation with RpDIMER_LIFETIME = 4 seconds.*

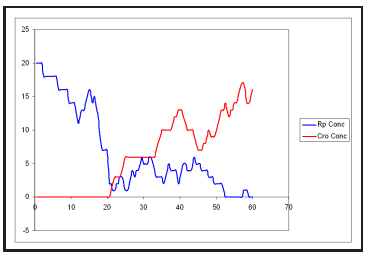


FIG. 4.7. *Rp vs. Cro, Rp_TO_OR1_TIME = 2, and Rp_TO_OR2_TIME = 2.*

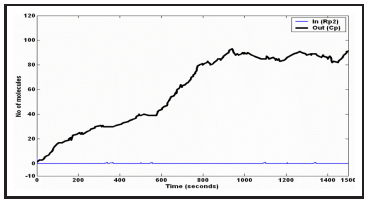


FIG. 4.8. *Stochastic modeling and simulation of Rp vs. Cro [13].*

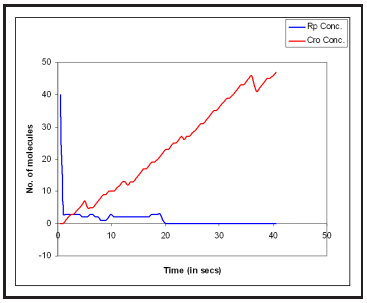


FIG. 4.9. *Simulation using parameters of [13] and agent-based modeling.*

TABLE 4.2
*The parameter values for simulation of phage lambda system for comparison with [13]*

| | |
|---|---|
| RPMONOMER_COUNT | 50 |
| RECA_COUNT | 100 |
| RECA_INIT_TIME | 0.5 |
| RPMONOMER_LIFETIME | 20 |
| CROMONOMER_LIFETIME | 50 |
| RECAMONOMER_LIFETIME | 60 |
| RP_TO_OR1_TIME | 1 |
| RP_TO_OR2_TIME | 2 |
| RP_TO_OR3_TIME | 3 |
| RPDIMER_LIFETIME | 5 |
| CRO_TO_OR1_TIME | 3 |
| CRO_TO_OR2_TIME | 2 |
| CRO_TO_OR3_TIME | 1 |
| CRODIMER_LIFETIME | 5 |
| RPDIMER_VEL | 20 |
| CRODIMER_VEL | 20 |
| RECA_VEL | 60 |
| RPMONOMER_VEL | 5 |
| CROMONOMER_VEL | 20 |
| RNA_VEL | 5 |

REFERENCES

[1] G. K. ACKERS, A. D. JOHNSON AND M. A. SHEA, *Quantitative model for gene regulation by λ phage repressor*, PNAS, 79(4) (1982), pp. 1129–1133.
[2] ANY LOGIC. http://www.xjtek.com Accessed June 15, 2007.
[3] S. ARBESMAN, *The chai-calculus: A computational formalism and its relationship to biological processes*, available online from http://www.santafe.edu/education/reu/2003/files/arbesman.pdf Accessed June 15, 2007.
[4] E. BONABEAU, *Agent-based modeling: methods and techniques for simulating human systems*, PNAS, 99(Suppl. 3) (2002), pp. 7280–7287.
[5] G. BOOCH, *Object-Oriented Analysis and Design with Applications*, Addison-Wesley, Reading, MA, 1994.
[6] G. BOOCH, J. RUMBAUGH, AND I. JACOBSON, *The Unified Modeling Language User Guide*, Addison-Wesley, Reading, MA, 1999.
[7] N. CANNATA, F. CORRADINI, E. MERELLI, A. OMICNI, AND A. RICCI, *An agent-oriented conceptual framework for systems bioilogy*, in Transactions on Computational Systems Biology III, Lecture Notes in Comput. Sci. 3737, Springer-Verlag, Berlin, 2005, pp. 105–122.
[8] M. DINSOREANU, I. SALOMIE, AND K. PUSZTAI, *On the design of agent-based systems using UML and extensions*, in Proceedings of the 24th International Conference on Information Technology Interfaces, 2002, pp. 205–210.
[9] P. FRYER, *What are complex adaptive systems?*, available online from http://www.trojanmice.com/articles/complexadaptivesystems.htm Accessed June 15, 2007.
[10] F. JACOB AND J. MONOD, *Genetic regulatory mechanisms in the synthesis of proteins*, J. Mol. Biol., 3 (1961), pp. 318–356.
[11] S. KHAN, R. MAKKENA, F. MCGEARY, K. DECKER, W. GILLIS, AND C. SCHMIDT, *A multi-agent system for the quantitative simulation of biological networks*, in Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems, 2003, pp. 385-392.
[12] J. KLEIN, *BREVE: a 3D environment for the simulation of decentralized systems and artificial life*, in Proceedings of Artificial Life VIII, the 8th International Conference on the Simulation and Synthesis of Living Systems, 2002, pp. 329–335.
[13] R. KRISHNAN AND C. PURDY, *Bio-inverter model and interface to digital hardware*, in Proceedings of the 48th IEEE International Midwest Symposium on Circuits and Systems, 2005, pp. 766–769.
[14] P. MENDEZ, *GEPASI: a software package for modelling the dynamics, steady states and control of biochemical and other systems*, Comput. Appl. Biosci., 9(5) (1993), pp. 563–715.
[15] D. MESTIVIER, P. Y. BOELLE, K. PAKDAMAN, A. RICHARD, J. P. COMET, G. HUTZLER, C. KUTTLER, A. IARTSEVA, AND F. KEPES, *Modeling of λ phage genetic switch*, available online from http://shum.huji.ac.il/ sorin/ccs/alex/ecoleThematiqueMontpellier2005.pdf Accessed June 15, 2007.
[16] N. MINAR, R. BURKHART, C. LANGTON, AND M. ASKENAZI, *The Swarm Simulation System: A Toolkit for Building Multi-agent Simulations*, Working Paper 96-06-042, Santa Fe Institute, Santa Fe, NM, 1996, available online from http://swarm.org/images/b/bb/MinarEtAl96.pdf Accessed June 15, 2007.
[17] E. NAMBOODIRI, P. HARTEN, AND C. PURDY, *Agent-based modeling of environmental systems*, in Proceedings of the Midwest Artificial Intelligence and Cognitive Science Conference (MAICS), 2005.
[18] M. PTASHNE, *A Genetic Switch: Phage λ and Higher Organisms*, 2nd ed., Cell Press: Blackwell Scientific Publications, Cambridge, MA, 1992.
[19] M. RESNICK, *StarLogo: an environment for decentralized modeling and decentralized thinking*, in Proceedings of the Conference on Human Factors in Computing Systems, 1996, pp. 11–12.
[20] S. M. SANCHEZ AND T. W. LUCAS, *Exploring the world of agent-based simulations: simple models, complex analyses*, in Proceedings of the 34th Conference on Winter Simulation: Exploring New Frontiers, 2002, pp. 116–126.
[21] G. TSUI, *λ-Switch–An Agent-based 3D Simulation*, CPSC502 Project Report, University of Calgary, 2002.
[22] K. WEBB AND T. WHITE, *Cell modeling using agent-based formalisms*, in Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, 2004, pp. 1190–1196.
[23] R. WEISS, G. HOMSY, AND T. F. KNIGHT, JR., *Toward in-vivo digital circuits*, in Dimacs Workshop on Evolution as Computation, Princeton, NJ, January 1999.
[24] M. WOOLDRIDGE AND N.R. JENNINGS, *Intelligent agents: theory and practice*, Knowl. Eng. Rev., 10(2) (1995), pp. 115–152.

*Edited by:* Dazhang Gu
*Received:* Jan 16, 2007
*Accepted:* April 15, 2007