I.J. Intelligent Systems and Applications, **2012, 2, 58-65**
Published Online March 2012 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2012.02.07

Agent-based Models in Synthetic Biology: Tools for Simulation and Prospects

E.V.Krishnamurthy
Australian National University,
Canberra, ACT 0200, Australia.
Evk.Krishnamurthy@anu.edu.au

Abstract- We describe a multiset of agents based modeling and simulation paradigm for synthetic biology. The multiset of agents –based programming paradigm, can be interpreted as the outcome arising out of deterministic, nondeterministic or stochastic interaction among elements in a multiset object space, that includes the environment. These interactions are like chemical reactions and the evolution of the multiset can emulate the system biological functions. Since the reaction rules are inherently parallel, any number of actions can be performed cooperatively or competitively among the subsets of elements, so that the elements evolve toward equilibrium or emergent state. Practical realization of this paradigm for system biological simulation is achieved through the concept of transactional style programming with agents, as well as soft computing (neural- network) principles. Also we briefly describe currently available tools for agent-based-modeling, simulation and animation.

Index Terms - Agent-based model, biological cell, chemical reaction model, motifs, simulation, Tools

I. INTRODUCTION

Biological Cells live in a very complex environment and receive many different signals- physical chemical, biological- and they need to monitor and respond to these signals within a reasonable time for survival and growth, by producing appropriate proteins on demand. The cell use special proteins called "Transcription factors" (TF) to internally represent the set of external environmental states. TF can be active or inactive and they can bind to the DNA to change the transcription rate of specific target genes, the rate at which mRNA is produced. These proteins act on the external environment and internal structure. Some proteins themselves act as TF that can activate or repress other genes. The above aspect of cellular cognition is analogous to the behavior of a multiset of agents carrying out concurrent transactions which besides being computational are also demand driven monitoring devices, as explained below.

The multiset of agents –based programming paradigm, can be interpreted as the outcome arising out of deterministic, nondeterministic or stochastic interaction among elements in a multiset object space, which includes the environment. These interactions are like chemical reactions and the evolution of the multiset can mimic the biological evolution. Since the reaction rules are inherently parallel, any number of actions can be performed cooperatively or competitively among the subsets of elements, so that the elements evolve toward an equilibrium or an emergent state.

Practical realization of this paradigm is achieved through the concept of transactions. Hence, this paradigm is widely applicable to all conventional algorithms, evolutionary algorithms, genetic algorithms, neural networks, self-organized criticality and active walker models (swarm and ant intelligence), DNA computing and modeling living systems. Further, the occurrence of motifs in systems biology, pioneered by Uri Alon [3]- such as auto-regulation, single input multi-output mode (SIM), dense overlapping regulons (DOR), Bifan, Feed forward loop (FFL), combinatorial logic and perceptron – like behavior, can be directly simulated using agents. Based on this analogy we can model several other systems biology aspects to synthesize biological systems. In this sense we can simulate in silico many synthetic biological problems, [5],[14].

In Section II we briefly state Ulon's [3] approach to motif-based systems biology. Section III describes an agent-based paradigm and its role in modeling, simulation and animation. In Section IV we describe how Motif based approach can be used for growing special purpose Genetic regulatory networks with specified goals, based on Genetic programming [3,9]. Bioinspired computing, Synthetic biology, Artificial life have all common aims realizing collective intelligence and natural computing They all use common tools: rule based systems, classifiers, Monte-Carlo, Simulated annealing, Genetic/Evolutionary algorithms. In Section V we briefly describe currently available tools for agent-based modeling and simulation. Section V1 contains the conclusion.

II. THE MOTIF APPROACH

The motif approach was pioneered by Uri Alon [3] and his colleagues. In a very lucid book, Alon [3] has examined transcription, transduction and communication networks in Systems Biology and has found that the following primitives are universally present: Feed-forward loops (FFL)- both coherent and incoherent, autoregulation (single loop), Multi-output FFL, Bifan. These motifs are very well suited for agent- based simulation (ABS), using transactional style programming and soft-computing principles. This approach will be useful from the point of view of understanding the basic aspects of cognition in cells to start with.

III. MULTI-SET OF AGENTS PARADIGM

The AOIS (Agent-Oriented System Community) defines an agent as a system that is capable of perceiving events in its environment or representing information about the current state of affairs and of acting in its environment guided by perceptions and stored information, Woolridge [30], Lucena et al. [18]. The multiset-agent system consists of several single agent-systems . Murthy and Krishnamurthy [19]. Thus if N agents are involved:
i = 1,2, , , N, each of the agents will be denoted with a label (i). Here, we will restrict ourselves to the definition that the agent is a software module having the transactional programming model [16] with all the important features- such as: atomicity of commitment, short duration transactions, sensing and aborting, and its own thread of control. 'This ensures that the process of matching and the follow-up actions satisfy the four important ACID properties: Atomicity (indivisibility and either all or no actions or carried out), Consistency (before and after the execution of a transaction), Isolation (no interference among the actions), Durability (no failure). Once all the actions are carried out and committed the next set of conditions are considered.
As a result of the actions followed by commitment, we may revise or update and obtain a new database for each agent; this may satisfy new conditions of the text and the actions are repeated by initiating a new set of transactions. These set of transformations halt when there are no more transactions executable or the databases does not undergo a change for two consecutive steps indicating a new consistent state of the databases.
However, if the interaction condition holds for several disjoint subsets of elements in the database at the same time, the actions can take place independently and simultaneously. This leads to cooperative parallelism; e.g. vector parallelism, pipeline parallelism.

A, Relation to Turing Machine taking advice

The agent - based paradigm is capable of solving problems that can be solved by Turing machine taking advice. This behavior arises due to interaction among the agents and the environment, endless communication and computation among agents and possibility for auto regulation. Such a system contains many cycles and the metric entropy (Lyapunov exponent) is positive-See Murthy, Krishnamurthy [19]. Such a system performs interactive- nonuniform computation with multiple inputs and outputs, and varying memory and instructions. Hence they are as powerful as evolving machines, as described in the works of Van Leewen , Wiedermann , Wegener, see Sekanina [23]. Note that Natural computing has the creative ability to upgrade its system whenever, and wherever required thus corresponding to the Turing machine taking advice and hence such a machine is more powerful to solve problems beyond Turing computable limits. The interactive agent paradigm provided with rule basis and performing nonuniform computation is thus a very powerful modelling tool.

B. Features of agents transactional paradigm

The multiset of agents paradigm (MAP) consists of the following features Murthy and Krishnamurthy [19].
(i) A multiset M that contains evolving agents (called the agent-space) whose information is structured in an appropriate way to suit the problem at hand. Unlike a passive data structure each agent is capable of autonomous computing.
(ii) A set of interaction rules that prescribes the context for the applicability of the rules to the agents. Each rule consists of a left-hand side (a pattern or property or attribute) describing the conditions under which the agents can communicate and interact, and a right hand side describes the actions to be performed by the agents, if the rule becomes applicable, based on deterministic, fuzzy or probabilistic criteria, Murthy and Krishnamurthy [19].
(iii) A control strategy that specifies the manner in which the agents will be chosen and interaction rules will be applied, the kinetics of the rule- interference (inhibition, activation, diffusion, chemotaxis) and a way of resolving conflicts that may arise when several rules match at once.
(iv) An agent by itself or another coordinating agent evaluates the performance of the agents to determine the effectiveness of rule application.
(v) Interaction -Based: The computations are interpreted as the outcome of interacting agents to produce new agents (or same agents with modified attributes) according to specific rules. Hence the intrinsic (genotype) or genetic constitution of an organism and acquired properties due to interaction (phenotype) with the environment can both be incorporated in the agent space. Since the interaction rules are inherently parallel, any number of actions can be performed cooperatively or competitively among the subsets of agents, so that the new agents evolve toward an equilibrium or unstable or chaotic state.
(vi)Content-based activation of rules: The next set of rules to be invoked is determined solely by the contents of the agent-space, as in the context of chemical reactions.
(vii) Pattern matching: Search takes place to bind the variables in such a way to satisfy the left hand side of the rule. It is this characteristic of pattern (or attribute)

matching that gives the agent-based paradigm its distinctive capabilities for nature-inspired computing.

(viii) Choice of objects, and actions:

We can several types of objects, as the basic elements of computation to perform suitable actions on them by defining a suitable topology, geometry or a metric space.

C. 1nteraction among Agents

In order to use the multi-agent paradigm to realise cooperative and competitive computational tasks, we need to consider how the agents can interfere with each other.

1. Enabling dependence (ED): Agent A(i) and agent A(j) are called enable dependent (or dataflow dependent) if the messages from A (i) creates the required precondition in A(j) to carry out a specific action .These messages can be chemical concentration, voltages or communication messages.

2.Inhibit dependence (ID): Agents A (i) and A (j) are called inhibit dependent, if the actions of A (i) creates the required precondition in A(j) to prevent it from executing a specific action. Inhibition may be due to negative weights in connective links or stop signals or other computable entities.

3. INTRAN Conflict (IC) : Agents A (i) and A (j) are opposition dependent (also called data-output dependent) through A(k)), if the order in which A (i) and A (j) enable A(k) and update A(k) produce different results in A(k); that is the objects A(i) and A (j) perform operations on A(k) that are not order reversible. That is, local serializability (or commutability) is not ensured in the INTRAN within A(k), if the actions are carried out within an agent in different partial order.

4. EXTRAN Conflict (EC): Agents A (i) and A(j) are data antidependent through A(k) if the order in which A(i) enables (inhibits) A(k), and A(j) enables (inhibits) A(k) result in different external actions (EXTRAN) by A(k) on the environment. That is the temporal order in which information arrives from the environment and other agents affects the global serializability (commutability) of the actions of an agent.

Remark: ED and ID:

The two properties ED and ID are crucial for modelling any life-like system which requires both positive and negative regulation and molecular switches. These rules permit an agent to enable itself (autocrine signalling for development or for autocatalysis), and also an agent A(i) to enable A(j) and A (j) to enable A(i) cyclically.

For example, A(i) can create the required precondition in A(k), so that A(j) can enable A(k). Also, A(i) can inhibit the required precondition in A(k) so that A(j) is prevented from enabling A(k).

Example: In cellular signal processing, proteins whose conformation can be altered by the equilibrium binding of a regulatory ligand switches between active and inactive states when the concentration of the regulator varies. Also situations such as- two ligands whose binding sites in a protein are coupled can reciprocally affect each others binding or proteins that can simultaneously bind two ligands - can be modelled by ED and ID.

Note that various types of Petri net models, are used currently in Systems Biology, Cardelli [6], Pinney et al.[22], Effroni et al [8], Harel [11], Aviv and Shapiro [4]. The agent models are more general and provide greater flexibility and also help in animation [25].

D, Concurrency and Conflicts

In distributed computing and transaction processing: we require that the following two conditions are satisfied for global serialization when concurrent operations take place.

1. At each agent the actions in local actions are performed in the non-conflicting order (Local serializability or commutativity).

2. At each agent the serialization order of the tasks dictated by every other agent is not violated. That is, for each pair of conflicting actions among transactions p and q, an action of p precedes an action of q in any local schedule, if and only if, the preconditions required for p do not conflict with those preconditions required for execution of the action q in the required ordering of all tasks in all agents (Global serializability).

The above two conditions require that the preconditions for actions in different agents A(i) and A(j) do not interfere or cause conflicts. These conditions are necessary for the stabilization of the multi-agent systems that the computations are locally and globally consistent.

Termination: For the termination of agent –based program, the interaction among the agents must come to a halt. When the entire set of agents halt we have an equilibrium state (or a fixed point) also called stability while dealing with exact computation in a deterministic system.

Non–termination, instability, multiple equilibria and chaos: These cases arise when the agents continue to interact indefinitely as in chemical oscillations, biological reactions, game theoretic problems, and cellular signal processing. Then the multiagent-system can have several attractors or equilibrium states. It is also possible that the evolution of the agent system is sensitive to initial conditions leading to chaos and self-organization.

Conflicts: Resolution or compromise? In agent -based modelling of behaviour, under concurrency, the conflicts arising in INTRAN and EXTRAN may require resolution or to an agreeable compromise. This situation can arise in animation and robotics. For example, the actions, "fold your right arm" or "stretch your right arm" are conflicting concurrent actions. We can achieve a compromise by keeping the arm folded half –way. Alternatively, one can blend the behaviour of actions, e.g., as in the actions "walk" and "run", if the quantitative parameters can be suitably averaged over. These rules should be based on the problem domain.

E. Agent-based Graph Motifs

Some of the basic motifs that occur [3] in abundance in many different biological networks have the following functions:

Enabling, Inhibiting, Cycles, Fork, Bifan, Diamond, Feed-forward loop, Feedback loop, Regulated feedback Autocatalysis, Bi-stable, Oscillator ,Repressilator

(Cyclically linked inhibitors. This occurs in production of proteins).

To describe all the above functions, we require temporal logical connectives such as: before, meets, overlaps, starts, during (contained in), finishes, coincident or their " inverse relations" after, met by, overlapped by, started by, contained by, finished by, coincident and also the relative timing of events. Note that the order of execution is necessary as illustrated in the following examples of temporal description of events occurring in the motifs. Hence, in practice, we need to use the concurrent programming (or multithreading) style of transaction processing to handle these aspects effectively, with a built in deterministic, non-deterministic or stochastic control. The transaction processing approach among interacting agents permits concurrent or partial order or total order execution with suitable time synchronization, as well as, in animation, [19].

F. Temporal (partial, total) order consistency

Agents can check for temporal consistency among events, synchronize them and can vary the temporal order of execution, if the time intervals are suitably assigned in their rule basis, with transactional style programming using locks or timestamps. For example agent A can enable agent B, which can delay its output to the next agent C in a deterministic, or probabilistic manner or non-deterministically delay until the agent C is in a position to receive this message-based on demand driven principle. Also agents can carry out last in first out order (LIFO) of execution and first in first out order (FIFO) of execution.

G. Production, Demand and Consumption rule

The Savageau rule [3], states that "A biosystem generates an amino acid, if that amino acid is not commonly available in the environment ". That is the system produces amino acids that are demand driven- the higher the demand and the less it is available in the environment, the system produce it more. This is realised by the producer-consumer model using interacting agents thus:

1. A monitoring agent checks the availability of the required amino acid in the environment. It enables the producer agent to produce the required amino acids.
2. There is a producer agent that generates amino acids and puts them in a buffer agent.
3. There is a consumer agent that consumes the amino acids from the buffer agent.
4. The buffer agent checks the consumption of amino acids from the buffer. If the buffer is full it inhibits the producer agent; if the buffer is empty, it enables the producer agent to produce. If the buffer is partly full it optimises the rate of consumption and production.

H. Special Purpose Agents in Simulation

Agents can be designed to realize special purpose functionalities. In building cellular biological models based on agents, it will be more convenient to use special purpose agents –such as sensor agents, goal agents, skill agents and compose these modules to build reusable, composable, blendable behavior. In this context, we may recall that in cell system biology, the protein networks organize themselves into special purpose proteins thus: Alberts et al [2], Cardelli [6], Wooley and Lin [29], Lauffenberger et al. [17].

1. Relay proteins: simply pass the message to the next signalling component
2. Messenger proteins: carry the signal from one part of the cell to another.
3. Adaptor proteins: Link one signalling protein to another
4. Amplifier proteins: Increase the signal levels causing a signal cascade
5. Transducer proteins: Convert the signal from one for to another
6. Bifurcation protein: Spread the signal from one pathway to another
7. Integrator proteins: Receive two or more signals and integrate them and transmit them.
8. Latent gene regulatory proteins: These are activated at the cell surface by activated receptors and then migrate to the nucleus to stimulate gene transcription.
9. Modulator proteins: These modify the intracellular signalling proteins and thereby regulate the strength of the signal along the pathway.
10. Anchoring proteins: Maintain a specific signalling proteins at a precise location in the cell by tethering them to a membrane of the cytoskeleton.
11. Scaffold proteins: These are adaptor or anchoring proteins that bind multiple signalling proteins together in a functional complex and often hold them at a specific location.

In animating protein -based computing, it is required to perform simple self-assembly, in which each agent has one or more binding sites to which ligands (a molecule that binds to a specific site on a protein) can bind , when they have complementary matching shapes. Once they match they can either inhibit or enable the action of the agents deterministically or stochastically. Signalling proteins can bind to one another in multiple combinations like Lego-bricks with the protein forming a 3-Dimensional network of interactions that determines the route followed by the signalling pathways. Agent-based animation can be helpful to understand specific sequence of reactions. Also the above special purpose agents can be useful.

I. Agents and Avatars in Animation

Agents consist of information objects and an associated script that knows what to do with the information and how to deal with the environment. They behave like actors and have intentions and actions. Agents are autonomous and they have a built in control to act, only if they want to. In addition, agents are flexible, proactive and have multithreaded control. The above aspects make agents suitable for reactive animation. In reactive animation, Efroni et al.[8], Harel [11] the system has to react to various kinds of events, signals and conditions that are often distributed and concurrent. Also they can be time critical exhibiting both digital and analog (or hybrid) behaviour. In addition the reactive system, as in cell

biological system can contain components that signal each other and also repeatedly created and destroyed.
Expansion of simulation to animation can be useful for teaching purposes -for example, in cell biology in which the molecules are involved in various interactions with other cells through cell-signalling cascades.
Cell signalling cascades consist of several steps of activation and inhibition, Alberts et al. [2]. These are usually illustrated in static form as a series of steps. Animation can introduce each step individually and in order to emphasize multiple effects of one protein and the cellular location of each effect. The animation emphasises the partial order of events, the binding of ligands and the accompanying reactions and the movements along with the timing of the events. Ideally, such a realization would require a total manifestation of the real object and the events that are occurring in time and the reactions of the associated objects. This can be achieved through the introduction of "Avatar" which is realised as an extension of the Agent model by including a body with a desired geometry and related functions. The body provides for the visualization part of the events. A scheduler that is controlled by one or more agents carries out the actions or animation on the body. Thus Avatar provides a visible part of the environment; it cannot perceive or react to events on its own.

J. Percolation

In many modelling situations involving particles (for example, when "Help yourself and suppress thy neighbour"(competitive policy), or attraction and repulsion dependencies are used. These dependencies can be modelled using a combination of Enable (Activation) and Inhibit dependence rules [3].

Also percolation-like situation (cooperative transition, in which the sharpness of response increases (decreases) with an increase in the number of effector (inhibitor) molecules that must bind to a target molecule can be modelled, Alberts et al [2]. In interactive animation of biological molecules static illustrations are not satisfactory. Transactional agents can enable us to animate the binding of certain molecules, accompanied by other actions. In particular, cell signalling pathways that involve inhibitory, binding, and activation steps can be animated without the inherent ambiguities arising in the order, timing and the interpretation of the events.

K. *Examples from System Biology*

In systems biology, the components are chemical concentration- based, and the activation of any component is based on the threshold of chemical concentration, as well as, its rate of change. In fact the activation function that results from two other concentrations is a weighted sum. This activation function when subjected to a threshold can decide whether the ultimate outcome is anactivation or inhibition .In fact a given motif can compute various logical functions or other functions depending upon the weights and the threshold. To illustrate this we consider examples of simple one layer and two layer neural networks.

It is well-known that a neural network consists of an input set of variables usually real numbers. These inputs are then multiplied by weighting functions and added . The resulting sum is then compared against a threshold function t; if the sum is greater than or equal to t the out put is one ; otherwise zero. A single layer neural network consists of a single set of inputs, weighting functions and summing device and a threshold function as described above. If these arrangements are repeated at several level it is called a multilayer neural network.

For example, if a threshold neural logic gate receives inputs x1 and x2 and if these are weighted by multipliers a1 and a2 respectively and added to form y= a1.x1+a2.x2, then the functions computed by this logic gate for the following cases, using the threshold function t(y) will be analysed below. We assume that by $0 \leq x1 \leq 1$, and $0 \leq x2 \leq 1$. The output region of these networks depend on the choice of the parameters (weights) a1 and a2. These weights along with the choice of the threshold play a role in determining the activation function.

Case 1: x1 and x2 are either 0 or 1; a1=a2=0.7, If t (y) $\geq$ 0.8, the output is 1; otherwise the output is 0. This is logical AND.

Case 2: x1 and x2 are either 0 or 1; a1=a2= 0.1, If t(y) $\geq$ 0.1,the output is 1 ;otherwise the output is 0 . This is logical OR.

Case 3: x1 and x2 are either 0 or 1; a1= a2= - 0.5. If t(y) $\geq$ - 0.1 the output is 1 ;otherwise the output is 0. This is logical NOR.

Case 4:x1 and x2 are either 0 or 1; a1= 0.5, a2 = - 0.5 .If t(y) $\geq$ - 0.8 the output is 1; otherwise the output is 0. This is a logical NAND.

Case 5: x1 and x2 are either 0 or 1; a1= 0.5, a2 = - 0.5 .If t(y) $\geq$ 0.1 the output is 1; otherwise the output is 0. This is a logical function (x1 and Not x2).

Case 6: Restrict to a single input by setting: x1= 0
Then with a1= 0, x2 is 0 or 1; a2= - 0.5 , and the condition if t (y) $\geq$ - 0.1, then output is 1; otherwise 0. Then the outputs when x2=0 or 1 will give a logical NOT

Consider now a two layer neural network consisting of the following inputs and weights at the two-levels.

Inputs: (x1,x2), weights a1, a2, b1, b2, c1,c2 and summing over takes place thus:
y1= a1.x1+b2.x2; y2= b1.x1+a2.x2; and z = c1.y1+c2.y2.
Let x1, x2 can be either 0 or 1 for the following values of a1,a2.b1,b2,c1 and c2, and the following thresholds t(y1), t(y2), t(z).
a1=a2=b1=b2= 1 ; c1=0.6 and c2 = - 0.2

If t(y1)≥ 0.4 the output is 1 ;otherwise 0. If t (y2)≥ 1.2 , its output is 1 ; otherwise 0.
Also; t(z) ≥ 0.5 it outputs 1; otherwise 0.
The function computed is Exclusive OR function . This is because:

Input(x1,x2) = (0,0): Then y1= 0, y2= 0 and so output t(y1) =0, and output of t (y2)= 0.
Then z= 0 and its output is 0.

Input(x1,x2) = (0,1) : then y1=1, y2=1 and so output of t(y1)=1 and output of t(y2)=0, since t(y2) is 1.2.
Then z= 0.6-0.0 = 0.6 ≥ 0.5; so its output is 1.

Input(x1,x2) = (1,0): then y1= 1,y2=1 and so output of t(y1)=1, output of t(y2) =0 since t(y2) is 1.2.
Then z= 0.6-0= 0.6≥ 0.5. Hence output is 1.
Input(x1,x2) =Input (1,1): then y1= 2, y2= 2 and so output of t(y1)=1 , t (y2)= 1,
and so z= 0.6- 0.2= 0.4 < 0.5 and output is 0.

Thus an identically looking motif can generate various functions in various different ways, and the many layer networks can produce different outputs depending upon the chemical concentration levels and threshold functions. If the inputs and outputs are stochastic in nature, the functions computed are no longer easily determinable.

IV. MOTIF-BASED GROWTH FOR TUNING CIRCUITS

Recent advances in Cell biology reveals that Nature has always concentrated on devising special purpose computing elements having both the analog and digital features, and exploiting cooperative and competitive schemes using percolation like threshold phenomena, positive feedback leading to self-enforcement, nonlinearity, and short and long range signalling to facilitate interaction using disordered protein networks. For accomplishing any special task, the protein machines are interconnected through softwiring, and are temporally and spatially coordinated through linked processes to achieve maximal efficiency. This problem is solved using Genetic programming with motifs as the subgraphs in growing a graph.

Growing a graph until it reaches self-organized criticality through interaction is closely related to Genetic Programming (GP), Koza [15], Goldberg [9]. In GP each program construct is a tree constructed from tasks, functions and terminal symbols. Then we perform crossover and mutation by swapping program sub-trees leading to feasible programs, taking care of the nature and type of the task. These operations resemble Metropolis-Hastings-Monte-Carlo methods to create transitivity in a graph from a given node to a desired attractor node. The GP operations correspond to an ergodic move-set in the space of graphs with a given set of parameters and repeatedly generating the moves and accepting them with probability p or rejecting them with probability (1-p).

Suitable move-sets are: creation of new nodes, aging and annihilation of nodes, Mutation -movement of edges from one place to another, mating -swapping edges of the form (s,t),(u,v) to (s,u),(t,v), adding new edges based on a cost function. Such moves can create a phase transition (or percolation) to reach a global goal through successive local goals. An important aspect in GP is the fitness of the individual program generated locally and globally. In self-organization, ideally, one requires that the fitness is a self-awareness function i.e. the individual who does the work evaluates itself, ensuring that the global fitness is guaranteed. This is widely prevalent in Nature for activities such as: nest building (stigmergy), food searching (foraging). For specified goals or for varying modularity this approach can be used to achieve special purpose genetic regulatory systems [3]. This approach is similar to using neural network methods.

As stated in Alon [3] the biological networks are modularly organized. In this sense, the conventional GP techniques as described above cannot capture the manner of biological evolution , since the random switching of edges, can destroy modularity. Agent-based simulation can retain the modular structure and vary the required connections to achieve a required goal.

V. MULTI-AGENT ARCHITECTURE AND TOOLKITS

Shakshuki et al. [24],evaluate multiagent tool kits, such as: Java Agent development framework (JADE), Zeus Agent building toolkit and JACK Intelligent Systems. They consider Java support, and performance evaluation. The number of agents they consider is of the order of 32. For the implementation of the paradigm described here, further developments are needed in Agent technology, since we need a very large number of agents to simulate many real-life scientific applications.

Gorton et al [10] have evaluated agent architectures: Adaptive Agent architecture (AAA), Aglets developed by IBM, and the Java based architecture Cougaar. The paradigm described here is well-suited for implementing in Cougaar, a Java based agent architecture, since Cougaar is based on human reasoning. A Cougaar agent consists of a blackboard that facilitates communication and operational modules called plug-in that communicate with one another through the blackboard and contain the logic for the agent's operations. The use of blackboard and direct communication are useful for simulating the problems in Synthetic biology.

Many other recent developments include Repast, North et al [20], Einstein, Ilachinski, in [1], and other agent based software tools, Adamsky and Komosinski [1], Odell[21] .Repast is object oriented and has a discrete event scheduler, 2D visualization, and can model Monte carlo, Genetic algorithms, neural nets. It can be used with a variety of languages: java, C#, managed C++, Prolog etc and is available for several Platforms. Swarm Software is a mixture of Object oriented C and Java and can be very useful for swarming and related simulations. Yet another tool is Star Logo, in [1].

VI. CONCLUSION

We described a multiset of agent based on transactional approach for modeling and simulation in synthetic biology. The use of specialized agent based graph motifs will be useful for realizing specific genetic regulatory networks with fixed goals and modularly varying goals using genetic programming. Currently available software tools are briefly described. In order to simulate and study genetic regulation models further it is necessary to enhance the currently available agent-based software tools with bioinformatics tools. However, much work needs to be done to refine these models to realize biological evolution of even simple mutation problems, e.g., transferring DNA from one organism and express it in a different organism to introduce specific new features,[7,28],e.g., those involving skin color changes in animals. This will require the bioinformatics machinery to be integrated along with agent based paradigm to explore the systembiology in great details, [12,13,27].

The present best technology available is a blend of object oriented and agent-based programming approaches that aid simulation and animation. However, there are several difficulties here:

1.The computational power of the gene control system can be shown to be not less than the computational power of a Turing machine that takes advice. This is clearly seen from the manner in which activators function. They can arbitrarily recruit coactivators. A coactivator may work by linking the activator or by promoting the conversion of a close chromatin structure to a more open structure.

2. Activators can have a multitude of targets. This makes it difficult even for in vitro experiments. So in silico models are less likely to provide useful information.

3. Repressors also can recruit co repressors and they act again in unpredictable ways.

4. Many genes interact with different proteins (too numerous) to regulate transcription. Such a binding can take place thro cooperativity or allostery (shape matching). It is difficult to classify which type of binding is used in different contexts and the outcome.

5.There are major obstacles in understanding protein networks and their functions. First of all the definition of functions in biology is entirely different from computable or mathematical functions.

Three kinds of interrelated functions occur in biology:

(i) The biochemical function referring to the chemical activity, binding and catalytic property and conformational changes.

(ii) The cellular function that is context dependent with respect to a tissue or an organ.

(iii) The phenotypic function that determines the behavioral and physiological properties of an organism in its environment.

In language theory, the above three functional aspects are analogous to the syntax that deals with the structure and grammar of a sentence (the chemical structure, bonds and reaction), the semantics that assigns a meaning to a sentence (its role in cellular action), and thirdly the pragmatics of a language which deals with the usability of the language in successfully meeting its goals (survival of an organism). All these functions are nonunique and context dependent. That is, similar functions can result in different outcomes and different functions can result in similar outcomes due to their many- to one, and one- to many mapping properties. This is called multiple realizability and leads to the failure of reductionism.

In mathematical logic these functions correspond to higher-order logical functions that deal with the non-denumerable properties of functions of another function of yet another function, and hence turn out to be non-computable. In addition, proteins are multi domain structures, undergoing extensive conformational changes and rapidly evolve their folds through order-disorder transitions making the complexity of folding problems in proteins computationally very complex.

We may now ask: What is the most suitable computational model and associated programming language to understand biological systems?

The biological systems are so complex to understand through formal means. The reason being they are more powerful systems than the Turing machine and associated computational schemes. In particular we can show that the biological systems are at least as powerful as a Turing machine taking advice as observed from several real life examples. Their power also arises due to interaction with the environment and autonomous entities, because they can watch out for their own set of internal responsibilities. Furthermore, they are interactive entities that are capable of using rich forms of messages both through chemical and electrical signals and through object interactions (shape matching and cooperation, repression, inhibition and competition). These messages can support method invocation—as well as informing the components involved in particular events, to respond or react or create a new entity. The computer science until recently was concerned with Turing like machines and programming was based entirely on algorithmic structure, e.g., for and while loops, if, assign, repeat command, In recent years however, there are several new developments that permit interactive programming based on object oriented technology (Java programming language) and agent based technology that can interact with the environment (internet). However, there are a number of difficulties in directly using these models as the basis and associated programming paradigms since the biological systems are both event, action and content based and they behave in deterministic, nondeterministic and stochastic manner with analog, digital signals and chemical reactions through shape matching and cooperative behavior. Hence it is only possible to model a very limited range of the biological systems. Even in vitro simulations have failed to yield sufficient information about biological systems. Accordingly, we cannot yet claim that we are really in a suitable level of understanding how to model biological systems. However, the object technology seems to reflect some of the biological system concepts so that we can model the biological system as a mixture

of object-oriented and agent based environment. These technologies are yet to mature to design suitable programming languages and model the biological system.

REFERENCES

[1]Adamsky,A and Komosinski,M. (2006), Artificial life Models in Software, Springer, New York.
[2]Alberts,B et al. (2002),The Molecular Biology of the Cell, Garland Science, New York.
[3]Alon, U. (2000), An Introduction to Systems Biology, Chapman and Hall, London.
[4]Aviv,R and Shapiro,E.(2002) Cellular Abstractors: Cellular computation, Nature, Vol 419, 343.
[5]Boloni,L et al (2004) Software Engineering Challenges for mutable-agent systems, Lecture Notes in Computer Science, Vol.2940,pp.149-166, Springer Verlag, New York.
[6] Cardelli,L. (2005) Abstract Machines in Systems Biology, Springer Transactions on Biological Systems, Springer Verlag, New York.
[7]Clark, D.P., and Russell,L.D., Molecular Biology, Cache River Press, Vienna, Ill.,1997
[8]Effroni,S et al. (2005) Reactive animation: Realistic Modeling of Complex Dynamic Systems, IEEE Computer, January, 33-46.
[9]Goldberg,D.E..(1989) Genetic algorithms in search, optimisation and machine learning, Addison Wesley, Reading, Mass.
[10] Gorton,I. (2004) Evaluating agent Architectures: Cougaar, Aglets and AAA, Lecture Notes in Computer Science,Vol.2940, Springer Verlag, New York,264-274.
[11]Harel,D.(2003) A grand challenge for computing: towards full reactive modeling of a multicellular animal, EATCS Bulletin, http:// www. wisdom.weizmann.ac.il/ ~dharel / papers/grandchallenge.doc.
[12]Jacob,C. and Burleigh,I., Biomolecular swarms-an agent based model of the lactose operon, Natural computing, Vol. 3,pp.361-376, 2004.
[13].Jacob,C, Barbasiewicz,A,and.Tsui,G, Swarms and Genes : Exploring Lambda switch gene regulation thro Swarm intelligence, Proc. IEEE congress on Evolutionary Computation, 2006, Vancouver.
[14]Keele,J.W and Wray, J.E. (2005). Software Agents in molecular computational Biology, Briefings in Bioinformatics, Vol.6, No.5,December, 370-379.
[15] Koza,J.R (1999) Genetic programming III, Morgan Kaufmann, San Francisco.
[16]Krishnamurthy, E.V. and Murthy, V.K(1991) Transaction Processing, Prentice Hall, N.J
[17] Lauffenburger,D.A.and Linderman,J.L. (1993) Receptors, Oxford University Press, Oxford.
[18]Lucena, C et al.(2004) Software Engineering for Multi-agent Systems, Lecture Notes in Computer Science, Vol.2940, Springer Verlag, New York.
[19]Murthy, V.K and Krishnamurthy, E.V (2009)," Multiset of Agents in a Network for Simulation of Complex Systems", in Recent advances in Nonlinear Dynamics and synchronization (NDS-1) -Theory and applications, Springer Verlag, New York, 2009. Eds. K.Kyamakya et al.
[20]North, M.J and Burton, E.J.,(2006), Escaping the accidents of History: An overview of Artificial life modelling with Repast, in [1], 115-142.
[21]Odell,J.J,Objects and agents compared, J. Object technology,(2002) Vol.1, 41-53,May-June.
[22]Pinney,J.W et al. (2003) Petri net representations in systems biology, Biochemical Society Transactions,Vol.31, Pt 6.
[23]Sekanina,L. (2005),Evolvable components,Springer, New York.
[24]Shakshuki,E and Jun,Y(2004) Multi-agent development toolkits: An Evaluation, Lecture Notes in Artificial intelligence, 3029, Springer Verlag, New York, 209-218.
[25] Stith, B.J. (2004) Use of animation in teaching cell biology, Cell. Biology Education, Vol.3(3),Fall,181-188.
[26] Thomas ,R and, D'Ari,R, Biological feedback, CRC Press, Boca Raton, Florida, 1990.
[27] Vallurpalli,V and Purdy,C Agent based modeling and simulation of biomolecular reactions, Univ. of Cincinnati, 2006.
[28]Watson,J.D et al, Molecular biology of the Gene, Benjamin Cummings, San Francisco , 2008.
[29] Wooley,J.C and.Lin,H.C. (Eds.) (2005) Catalyzing inquiry at the interface of computing and biology, National Academies Press, Washington, DC.
[30] Woolridge,M(2002) Introduction to Multi-Agent systems, John Wiley, New York.

Professor E.V. Krishnamurthy is with the Computer Sciences Laboratory, Australian National University. He is the author of several books and papers in Computer Science and Information technology.

Address:
Computer Sciences Laboratory, Research school of Information Sciences and Engineering,Bldg.115 Australian National University, Canberra, ACT 0200, Australia.
Email: Evk.Krishnamurthy@anu.edu.au

How to cite this paper: E.V.Krishnamurthy,"Agent-based Models in Synthetic Biology: Tools for Simulation and Prospects", International Journal of Intelligent Systems and Applications(IJISA), vol.4, no.2, pp.58-65, 2012. DOI: 10.5815/ijisa.2012.02.07

