MICRORAM: A SIMULATION MODEL OF A COLONY OF BACTERIA EVOLVING INSIDE AN ARTIFICIAL WORLD

Daniel Thai Dam, Rafael Lahoz-Beltra

Abstract: *MICRORAM is a simulation model in which a colony of bacteria evolves inside an artificial world. The model has the flavor of the classical models of the decades of the 80s and 90s in which artificial life was inspired by microbiology. We show how a population of 'bacterial' agents is able to adapt to environmental changes and survive to the attack from an external agent simulated with an 'antibiotic'. The conclusion is that many ideas from the 80s and 90s are still valid, and it is possible to design and simulate agents inspired by natural 'bacterial colonies', with potential applications in bacterial and natural computing.*

Keywords: *artificial life worlds, agent based modeling, bacterial genetic algorithm, conjugation operator.*

ACM Classification Keywords*: I.6 Simulation and Modeling*

Introduction

MICRORAM, an abbreviation for "MICROORGANISMS living in a RAM memory", is a simulation model in which a colony of bacteria evolves inside an artificial world. The model has the flavor of the classical models of the decades of the 80s and 90s in which artificial life was inspired by microbiology [Lahoz-Beltra, 2004; 2008]. We show how a population of 'bacterial' agents is able to adapt to environmental changes and survive to the attack from an external agent simulated with an 'antibiotic'. This was the first model [Thai Dam, 1997] from many others were obtained [Lahoz-Beltra et al., 2014a; Perales-Gravan and Lahoz-Beltra, 2008; Perales-Gravan et al., 2013; Recio Rincon et al., 2014] so that this work has sentimental meaning. According to the prevailing ideas in those decades, the behavior of a *program* can be used to support a complex theory [Partridge and Lopez, 1984], being in some cases the program 'the only irrefutable proposition about such theory'. A good example of this idea is found in a computational model of the chemokinetic behavior in *Paramecium* [Van Houten and Van Houten, 1982]. The development of techniques well known today, as is the case of genetic algorithms and cellular automata, promoted the design of this class of models. On the one hand the ability to simulate the evolution with genetic algorithms leads to models such as the Dewdney's model [Dewdney, 1989] in which the evolution of a population of protozoa in a pond is simulated in a computer. On the other, cellular automata technique allowed in past decades simulate the growth and branching in fungi or conduct the simulation of bacterial colony growth using the BRANCH automata [Ermentrout and Edelstein-Keshet, 1993].

Despite the time elapsed in which were in vogue this class of models, we believe that the design of these models can still provide significant results in biology as well as in bacterial and natural computing [Lahoz-Beltra, 2012; Lahoz-Beltra et al., 2014a]. In the next sections we describe the general features of the MICRORAM model.

Model description

In this section we introduce MICRORAM, that is, the algorithm that results once conjugation as well as other biological features is included in a genetic algorithm. The MICRORAM model consists of three elements (Figure 1):

1. **Agents** - Agents are 'bacteria', capable of performing certain cellular functions, reproduce and evolve over time.

2. **Environment** - It is the artificial world in which the agents inhabit and whose features create a selection pressure. The environment is divided into cells, thus a 2D lattice, it is a finite space but unlimited to be a toroidal shape. We use the word **cell** to refer to these 'slices' of the environment.

3. **Genotype** - The agents have a chromosome being the conjugation the genetic mechanisms responsible for the variability.

The dynamic behavior of the model is due to the interaction of these three elements simultaneously being modified the *genotype* and the *environment* through the *agent*.

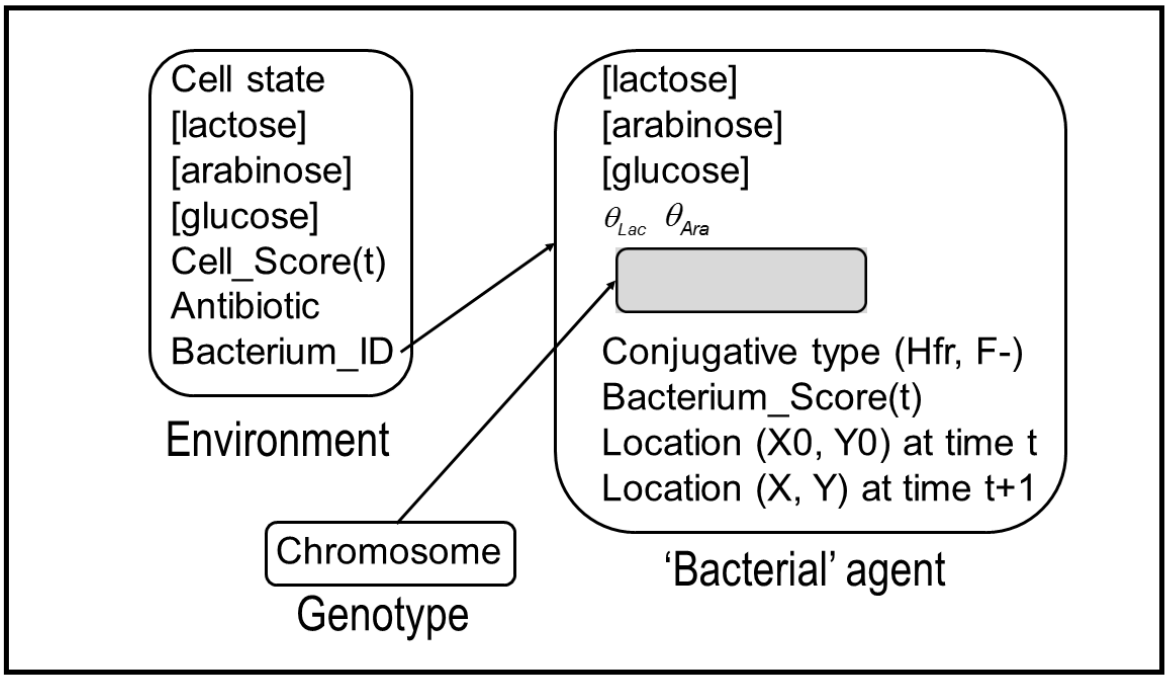

Figure 1. Architecture of a MICRORAM agent, the environment and genotype

Cellular functions of bacterial agents

A 'bacterial' agent performs the following biological tasks:

Nutrition - Bacteria need a source of *energy* (E) for survival, either from three sugars: glucose (E=1), lactose or arabinose (E=2). At the beginning of the simulation MICRORAM randomly assigns specific amounts of sugars in environmental cells and bacteria. The regulation of sugars is through a *model of operon* (cluster of genes under the control of a single promoter). Sugars in the grid cells are not replaced as they are consumed, producing this feature a Darwinian pressure. If the amount of a particular sugar is below a certain threshold then the sugar is not taken up by the bacteria. Furthermore, when glucose in bacteria is exhausted then the conversion of lactose or arabinose in two glucose molecules is triggered. The processes described above can be algorithmically defined as follows. Let $[\text{lactose}]_{\text{cell}}$, $[\text{arabinose}]_{\text{cell}}$, $[\text{glucose}]_{\text{cell}}$ be the amounts of lactose, arabinose and glucose in an environmental cell (Figure 1):

- Compare the value of $[\text{lactose}]_{\text{cell}}$ with threshold θ_{Lac} :

if [lactose] $_{cell}$> θ_{Lac} then activate the 'Lac operon' subtracting a lactose unit to cell register and adding a lactose unit to bacteria register.

if [lactose] $_{cell}$ <= θ_{Lac} then disable the 'Lac operon' not updating the lactose records.

- Compare the value of [arabinose]$_{cell}$ with threshold θ_{Ara} :

  if [arabinose] $_{cell}$> θ_{Ara} then activate the 'Ara operon' subtracting an arabinose unit to cell register and adding an arabinose unit to bacteria register.

  if [arabinose] $_{cell}$ <= θ_{Ara} then disable the 'Ara operon' not updating the arabinose records.

- Compare the value of [glucose]$_{cell}$ with threshold θ_{Glu} :

  if [glucose] $_{cell}$> θ_{Glu} then subtract a glucose unit to cell register and adding a glucose unit to bacteria register. Otherwise, not update the records of glucose.

  if [glucose] $_{cell}$<= θ_{Glu} then:

  if the bacteria have intracellular lactose then subtract a unit from lactose bacterial record and adding two glucose units in glucose bacterial record. Otherwise, records are not updated.

  if the bacteria have intracellular arabinose then subtract a unit from arabinose bacterial record and adding two glucose units in glucose bacterial record. Otherwise, records are not updated.

The adaptability of a bacterium, that is its *fitness*, is given by the amount of glucose accumulated (or equivalently *S*(t), the bacterial ***score***).

Motility - Bacteria move in two different ways. If a bacterium has flagellum (appendage that protrudes the bacterium which role is locomotion) [Lahoz-Beltra, 1997] then it will move attracted to cells with higher sugar content, otherwise it will move erratically. In the case bacterium has flagellum the draw of the cells is done by applying a 'biased roulette' method. Given the initial position (X0, Y0) of the bacteria and defining a Moore neighborhood only are selected those cells (X, Y) not occupied by bacteria. Bacterium location is updated synchronously in the population (thus, the bacterial colony). When a bacterium moves occurs an energy expenditure in the form of glucose. In the simulation experiments this value was set equal to E = 5.

Cell division - After a certain time, one bacterium is divided into two bacteria. Cell division is simulated as follows: daughter bacterium inherits the parameters of the mother (or alternatively, parental) *bacterium record* (Figure 1). The chromosome and the sugars thresholds are similar in both bacteria, the mother and daughter. However, the bacterial score and intracellular (or bacterial) sugars of the parent bacterium are divided equally between the two bacteria. Mother bacterium will stay in its environmental cell, and the new bacterium will occupy a neighboring cell. If there are no empty cells in the vicinity then bacteria will not be divided.

Bacterial chromosome - The chromosome of a bacterium is a string of 15 bits (Figure 2) such that the first two bits encode the (1-2 positions) Lac operon, the next two bits represent the (3-4 positions) Ara operon, 10 bits are for the antibiotic inhibitor gene (5-14 positions), and the last bit codes for the flagellum gene (15 position). The presence or absence of flagella depends on the value of the last bit encoding this organelle: if the bit value is 1 then the bacterium has flagellum. Otherwise, if the bit value is 0 then the bacterium does not have flagella.

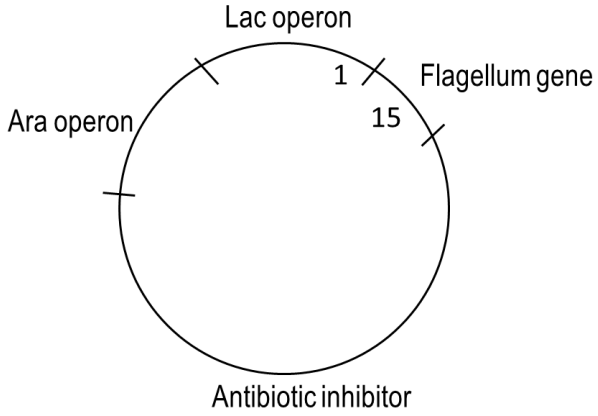

Figure 2. Bacterial chromosome

The operon model (Table I) was simplified to two bits (Figure 3): the first bit (bit controller or regulator) simulates the gene promoter and gene operator, the second bit simulates the structural gene. Its operation is similar to a switch with on/off states, such that a NOT operator is applied to the regulator bit: if the regulator bit is equal to 0 then the structural gene is activated to state 1, capturing sugar (lactose or arabinose) from the environment (environmental cell). Otherwise, no sugar is captured from environment. It is important to note that the operons are regulated by the environment. That is, the regulator bit is modulated by the presence of lactose or arabinose in the environment, activating the operons when the amounts of these sugars are above a threshold value.

Table I- Operon model

REGULATOR GENE	STRUCTURAL GENE	ACTION
0	1	bacterium captures sugar
1	0	bacterium does not capture sugar

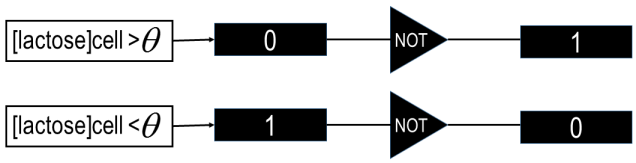

Figure 3. Lac operon model. Depending on lactose concentration in the environment a regulator gene is in 0 or 1 state. A NOT boolean operator is applied to regulator resulting the activation (1) or repression (0) of the structural gene. A similar model is used with arabinose

Antibiotic inhibitor - The antibiotic inhibitor gene is a string of 10 bits length which is coding for a peptide molecule. The peptide binds to an antibiotic molecule according to the key-lock principle. That is, the degree of coupling between the peptide and the antibiotic molecule is simulated according to the Hamming distance, such that 0 simulates a 'valley' and 1 simulates a 'peak' in the molecule conformation. A greater inhibition of antibiotic greater Hamming distance (for example, 1-0 means that a peak of the peptide is coupled to a valley of the antibiotic), surviving the bacterium. Therefore, in the MICRORAM model the degree of inhibition of the antibiotic is simulated through a coefficient k (Table II) that affects to the score of the bacterium: $S(t+1) = k\ S(t)$. Note that for a Hamming distance less than or equal to 2 the peptide does not inhibit the antibiotic being the $S(t+1)$ equal to 0, thus killing the antibiotic the bacterium.

Table II

HAMMING DISTANCE	***k***
0, 1, 2	0
3, 4	0.25
5, 6	0.5
7, 8	0.75
9, 10	1.0

Conjugation - Bacteria exhibit significant phenomena of genetic transfer and crossover between cells. This kind of mechanism belongs to a particular kind of genetic transfer known as horizontal gene transfer. Horizontal, lateral or cross-population gene transfer is any process in which an organism, i.e. a donor bacterium, transfers a genetic segment to another one, a recipient bacterium, which is not its offspring. Conjugation is one of the key genetic mechanisms of horizontal gene transfer between bacteria. In previous papers we introduced a bacterial conjugation operator showing its utility by designing an AM radio receiver [Perales-Gravan and Lahoz-Beltra, 2008] as well as a genetic algorithm including transduction [Perales-Gravan et al., 2013; Lahoz-Beltra et al., 2014b] and which we called as PETRI (*Promoting Evolution Through Reiterated Infection*). Indeed MICRORAM is a model that was the ancestor [Thai Dam, 1997] of this family of models, presented, years later, in this paper.

From a biological point of view we have considered only the simulation of the conjugation of the type Hfr x F-. However, this detail can be omitted without affecting the understanding of the model. If a Boolean variable takes the value True then bacterium is Hfr, being the bacteria F- when the value of the Boolean variable is False. The daughter bacterium inherits the conjugative type from mother bacterium. Conjugation is simulated as follows.

First, a pair of bacteria $\{i, j\}$ are selected according to the next algorithm. The Hfr bacterium searches for the F- bacterium in its neighborhood. In the event that the Hfr bacterium had several candidates randomly select its partner. Second, the transfer of the genome is simulated as follows. We have considered two genetic operators depending on where it is located on chromosome the insertion point of the F plasmid or fertility factor. This factor allows genes to be transferred from one bacterium carrying the factor to another bacterium lacking the factor (http://en.wikipedia.org/wiki/F-plasmid). Thus, we introduced two genetic operators, conjugation types I and II:

a) Type I:
    1. On chromosome of a (Hfr) donor bacterium, obtain the length l of the strand transferred to the recipient bacterium (F-). The l value has been simulated applying Monte Carlo's method and

assuming DNA lengths exponentially distributed with α parameter [Perales-Gravan and Lahoz-Beltra, 2008];

2. The transferred fragment is inserted into the chromosome of the recipient bacterium, replacing the chromosome segment, between positions 1 and *l*.

b) Type II:

1. On chromosome of a (Hfr) donor bacterium, obtain a random number *U* modeling the insertion location of the F plasmid and the length *l* of the strand transferred to the recipient bacterium (F-);
2. The transferred fragment is inserted into the chromosome of the recipient bacterium, replacing the chromosome segment, between positions *U*+1 and *l*.

For instance, consider a type II conjugation with *U*=2 and *l*=6 being the bacterial chromosome 'Lac operon – Ara operon- Antibiotic inhibitor – Flagellum gene' with gene values 10-11-0000000000-1. In consequence, the strand transferred to the recipient bacterium begins in position 3 and ends in position 8 being the 'DNA' fragment 11-0000. In this example, if the recipient chromosome was 00-00-1111111111-0 then after crossover the resulting chromosome will be 00-11-0000111111-0.

Third, the segregation is simulated retaining unchanged the Hfr donor bacterium, updating the chromosome in the recipient bacterium which retains the conjugative type F-. Conjugation operators I and II are similar to COFP (Conjugation Operator with a Fixed Point) and CORP (Conjugation Operator with a Random Point) described in [Perales-Gravan and Lahoz-Beltra, 2008]. The main difference is that the conjugation of type I and II are *bioinspired* operators designed with a *theoretical purpose*, whereas COFP and CORP have a *practical purpose* in the context of genetic algorithms [Davies, 1991; Goldberg, 1989].

Simulation experiments

Before studying the evolution of a bacterial colony or population we compared classical crossover operators [Davies, 1991; Goldberg, 1989] of a genetic algorithm with the bacterial recombination (thus, conjugation types I and II) operators proposed in this paper. Homologous one-point and two-points crossover operators were compared with the conjugation operator, choosing in all experiments a population size N = 100 bacteria, chromosomes with length *l* = 17 bits and $\alpha = 0.24$, being the mutation rate equal to 0.08. In the simulation experiments we study the optimization of the following objective function:

$$f(x)=x_{10} \tag{1}$$

where x_{10} is the value in base 10 of the chromosome or binary string x. The study was conducted for a total of 200 generations, conducting the simulation of two kinds of experiments. In a first set of experiments the *environment remains stable*, whereas in another batch of experiments we *changed the environment* after a certain number of generations. Given a certain number of generations, the change of environment was simulated replacing the objective function f(x) by f'(x):

$$f'(x)=131071-f(x) \tag{2}$$

The change of environment was simulated in the generation 5 or 25; the recombination rate was 25%, 50%, 75% and 100%. The simulation experiments performance was evaluated according to the statistical methods described in [Lahoz-Beltra and Perales-Gravan, 2010; 2014]. In MICRORAM model, the evaluation of genetic operators was conducted studying 10 replicate experiments (n), obtaining the average fitness ($\bar{f}$):

$$\bar{f} = \frac{\sum_{i=1}^{N} f(x_i)}{N} \tag{3}$$

and the standard error of the mean (SEM):

$$SEM = \frac{\sqrt{\frac{\sum_{i=1}^{N}\left(x_i - \bar{f}\right)^2}{N}}}{\sqrt{n}} \tag{4}$$

Concluded this preliminary study, we will describe the experiments carried out with MICRORAM simulator:

Experiment 1. First, we performed a set of experiments in which the bacteria were using homologous recombination instead of conjugation. The experiments were performed with antibiotic present in the environment, studying the antibiotic 'molecule' 1100110011. Experiments without the antibiotic were also conducted. All simulation experiments were performed at different recombination rates, population size N = 100, length of chromosome *l* =15 and mutation rate 0.08.

Experiment 2. Using the same experimental conditions described above, we conducted a batch of experiments in which the bacteria used the conjugation operators, types I and II.

It is important to note that in bacteria with homologous recombination, recombinant bacterium replaces the parental bacteria; while in the bacteria with conjugation, the colony retained both, thus the recombinant bacterium and parental bacterium (donor).

Results

The results of the simulations fit reasonably with observable results in a colony of bacteria, justifying the assumptions of the model and the parameters values. Figure 4 illustrates a sequence of states representative of a population of artificial bacteria simulated with MICRORAM. In Figure 5 we show the results obtained with classic crossover operators in a standard genetic algorithm, showing a convergence of the bacterial colony towards an average fitness value equal to 1200. The SEM value ranged from a 3 minimum to 10 maximum values. In the experiments carried out with conjugation type I no differences were obtained in the minimum SEM value, decreasing the maximum SEM value and therefore the maximum chromosomal variability. However, when the parental bacterium (donor) is removed from the colony then increases the minimum and maximum SEM values, and therefore increases the chromosome population variability (Figure 6a).

In experiments in which a change of environment was simulated the following results were obtained. What was observed when the change of environment occurs in the fifth generation? In conjugation type I experiments the fitness value decreased to [600, 800], except for the case with a recombination rate of 25% in which the population converged to a fitness value equal to 1200. Moreover, in this case the population converges in a stepwise fashion (Figure 6b), also decreasing in the same fashion the SEM value. The results obtained with type II conjugation resemble a standard genetic algorithm in which there is simulated a change of environment (Figure 6c). But what happens when the change of environment arises in generation 25? In this case and for the conjugation of type I with recombination rate of 25% the results are similar to those where the change of environment occurs in the fifth generation. However we highlight the case (Figure 6d) in which the bacterial colony after a decreases in chromosome variability between the 60 and 100 generations, experiences a sharp increases of variability which stabilizes from generation 120.

Finally, in the experiments performed in the presence of antibiotic the following results were obtained. In all experimental situations the antibiotic effect was to decrease the fitness of the population or bacterial colony. When bacteria use the classic one-point crossover operator or alternatively conjugation type I the optimal recombination rate was 75%, while the optimal recombination rate was 50% for the classic two-points crossover operator or conjugation type II.

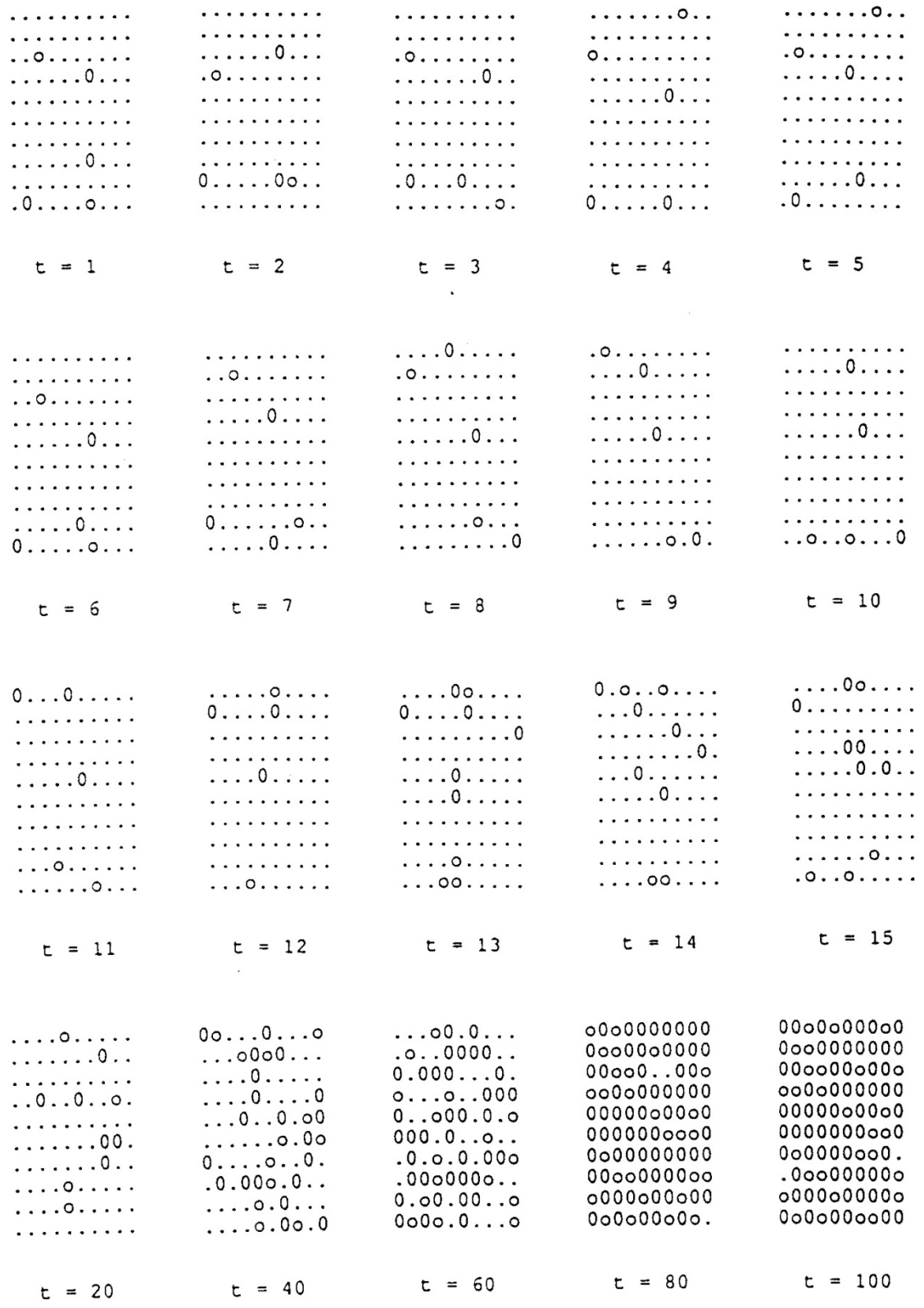

Figure 4. A colony of 'bacterial' agents evolving inside an artificial world (0=Hfr, o=F-)

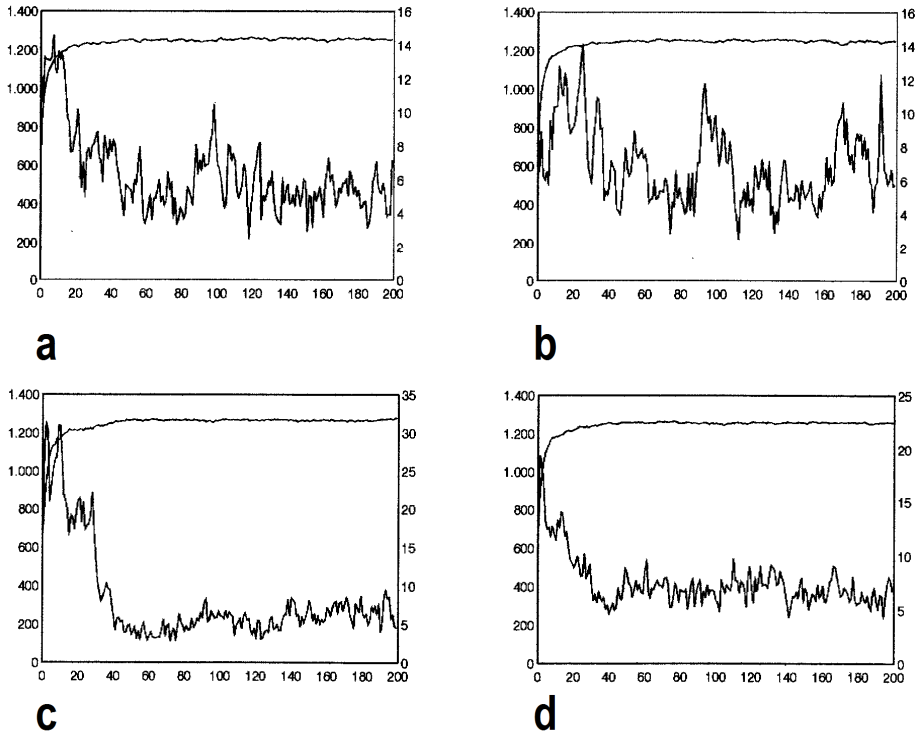

Figure 5. Performance plots showing the average fitness value (continuous line) and SEM (sawtooth curve) vs. generation time. (a) classic one-point crossover (b) classic two-points crossover (c) conjugation type I (d) conjugation type II.

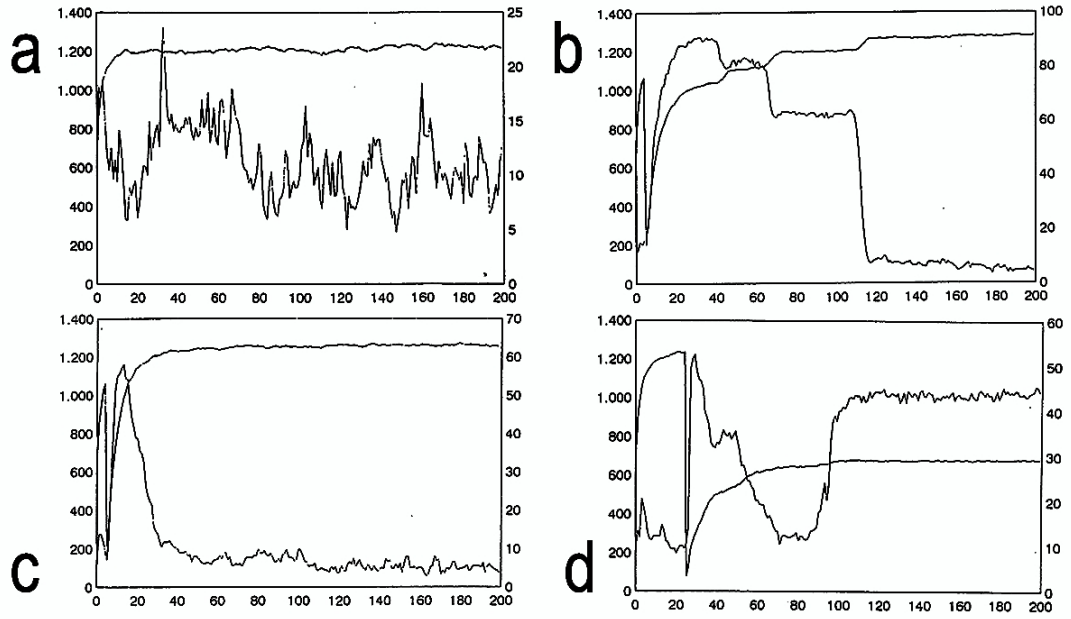

Figure 6. Performance plots showing the average fitness value (continuous line) and SEM (sawtooth curve) vs. generation time. Conjugation type I (a) when bacterium donor is removed and with a (b) change of environment in the 5th generation. (c) Conjugation type II with change of environment in the 5th generation (d) Conjugation type I with a change of environment in generation 25.

Conclusion

We modeled and simulated a population or bacterial colony based on an agent algorithm, exploring the role of conjugation in a genetic algorithm. The simulator has been named MICRORAM, showing how a population of 'bacterial' agents is able to adapt to environmental changes and survive to the attack from an external agent simulated with an 'antibiotic'. This was the first model [Thai Dam, 1997] from many others developed recently, so that this work has a 'sentimental meaning'. The conclusion is that many ideas from the 80s and 90s are still valid, and it is possible to design and simulate agents inspired by natural 'bacterial colonies', with potential applications in bacterial [Lahoz-Beltra, 2012; Lahoz-Beltra et al., 2014a] and natural computing [Recio Rincon et al., 2014].

Bibliography

[Davies, 1991] L. Davies. Handbook of Genetic Algorithms. Van Nostrand Reinhold.

[Dewdney, 1989] A. K. Dewdney. 1989. Evolución simulada: Un programa en que los microbios aprenden a cazar bacterias. Investigación y Ciencia 154: 96-100 (Spanish edition of Scientific American).

[Ermentrout and Edelstein-Keshet, 1993] G.B. Ermentrout, L. Edelstein-Keshet. 1993. Cellular automata approaches to biological modelling. Journal Theoretical Biology 160: 97-133.

[Goldberg, 1989] D. E. Goldberg. Genetic algorithms in search, optimization and machine learning. Addison-Wesley.

[Lahoz-Beltra and Perales-Gravan, 2010] R. Lahoz-Beltra, C. Perales-Gravan. 2010. A survey of nonparametric tests for the statistical analysis of evolutionary computational experiments. International Journal Information Theories and Applications 17: 49-61.

[Lahoz-Beltra and Perales-Gravan, 2014] R. Lahoz-Beltra, C. Perales-Gravan. 2014. Appendix. A survey of nonparametric tests for the statistical analysis of evolutionary computational experiments. figshare. http://dx.doi.org/10.6084/m9.figshare.1125796

[Lahoz-Beltra et al., 2014a] R. Lahoz-Beltra, J. Navarro, P.C. Marijuan. 2014. Bacterial computing: a form of natural computing and its applications. Frontiers in Microbiology Article 101. http://journal.frontiersin.org/Journal /10.3389/fmicb.2014.00101/full

[Lahoz-Beltra et al., 2014b] R. Lahoz-Beltra, C. Perales-Gravan, J. de Vicente Buendia, J. Castellanos. 2014. Appendix. Modeling, simulation and application of bacterial transduction in genetic algorithms. figshare. http://dx.doi.org/10.6084/m9.figshare.1125797

[Lahoz-Beltra, 1997] R. Lahoz-Beltra. Molecular automata assembly: principles and simulation of bacterial membrane construction. BioSystems 44(3): 209-229.

[Lahoz-Beltra, 2004] R. Lahoz-Beltra. 2004. Bioinformatica: Simulacion, Vida Artificial e Inteligencia Artificial (Transl.: Spanish). Ediciones Díaz de Santos, Madrid, Spain.

[Lahoz-Beltra, 2008] R. Lahoz-Beltra. 2008. ¿Juega Darwin a los Dados? (Transl.: Spanish). Editorial NIVOLA, Madrid, Spain.

[Lahoz-Beltra, 2012] R. Lahoz-Beltra. 2012. Cellular computing: towards an artificial cell. International Journal Information Theories and Applications 19: 313-318.

[Partridge and Lopez, 1984] D. Partridge, P.D. Lopez. 1984. Computer programs as theories in biology. Journal of Theoretical Biology 108: 539-564.

[Perales-Gravan and Lahoz-Beltra, 2008] C. Perales-Gravan, R. Lahoz-Beltra. 2008. An AM radio receiver designed with a genetic algorithm based on a bacterial conjugation genetic operator. IEEE Transactions on Evolutionary Computation 12(2): 129-142

[Perales-Gravan et al., 2013] C. Perales-Gravan, J. de Vicente Buendia, J. Castellanos, R. Lahoz-Beltra. 2013. Modeling, simulation and application of bacterial transduction in genetic algorithms. International Journal Information Technologies & Knowledge 7:11-22.

[Recio Rincon et al., 2014] C. Recio Rincon, P. Cordero, J. Castellanos, R. Lahoz-Beltra. 2014. A new method for the binary encoding and hardware implementation of metabolic pathways. International Journal Information Theories and Applications 21: 21-30.

[Thai Dam, 1997] D. Thai Dam. 1997. MICRORAM: Un modelo orientado a la simulación de la evolución de una población de bacterias artificiales. Tesina, Facultad de Biología, Universidad Complutense de Madrid. (Transl.: Spanish).

[Van Houten and Van Houten, 1982] J. Van Houten, J.C. Van Houten.1982. Computer simulation of Paramecium chemokinesis behavior. Journal Theoretical Biology 98: 453-468.

Authors' Information

Daniel Thai Dam *– Daniel Thai is a BS in Pharmacy and is currently working as Senior Site Support Engineer at Quintiles, e-mail: dThrak@gmail.com*

Interests: IT processes optimization & synchronization in messy environments

Rafael Lahoz-Beltra – *Department of Applied Mathematics (Biomathematics), Faculty of Biological Sciences, Complutense University of Madrid, 28040 Madrid, Spain, e-mail: lahozraf@ucm.es*

Major Fields of Scientific Research: evolutionary computation, embryo development modeling and the design of bioinspired algorithms