Transcription and Evolution of a Virtual Bacteria Culture

Ricardo M. Hoar
University of Calgary
2500 University Drive NW
Calgary, AB, Canada T2N1N4
hoarr@cpsc.ucalgary.ca

Joanne K. Penner
University of Calgary
2500 University Drive NW
Calgary, AB, Canada T2N1N4
pennerj@cpsc.ucalgary.ca

Christian Jacob
University of Calgary
2500 University Drive NW
Calgary, AB, Canada T2N1N4
jacob@cpsc.ucalgary.ca

Abstract- This paper describes implementation details and results from a simulated multi-agent bacteria ecosystem. Each bacterium is encoded on a DNA-like genome, specifying the genes and proteins it can produce. These compounds interact in an artificial chemistry, producing regulatory networks and observable behaviours such as moving, splitting and eating. By using an artificial chemistry to model gene regulation, artificial bacteria are able to live, reproduce and evolve.

1 Introduction

Bacteria are a testament to the effectiveness of evolution as a mechanism for adaptation and are able to thrive in nearly all earthly environments. However complicated the actual life of a bacterium is, it is simple compared to more complex organisms and thus is the basis for much study [2]. These simplest organisms will undoubtedly yield insight into the inner workings of all life, including ourselves.

In addition to the appeal of bacteria from a microbiology perspective, the interaction between bacteria is also of great interest. Research of social organisms such as ants and bees has shown that local interactions and decisions result in emergent collective intelligence [4, 6]. These multi agent paradigms have been applied to model decentralized, massively-parallel systems—from ant colonies to vehicle traffic [12, 5].

Through the simulation of bacteria, we are able to observe a similar, emergent order from the culture. By encoding observable traits of the bacteria and colouring each bacterium’s emissions, global effects can be visualized, adding an intuitive aspect to the analysis. Emission patterns produce seemingly creative images, illustrating evolution and inheritance patterns that exhibit self organization.

Facilitated by use of an artificial chemistry setup, we evolve gene regulatory networks. Our simulated bacteria live by updating their concentrations of molecules in each time step. Whenever molecules match user-defined patterns, observable behaviours (functions), such as splitting, can be seen. This paradigm allows for arbitrary gene regulation networks, which result in a large domain of possible individuals. By encoding the genotype of each bacterium as a virtual DNA sequence and interpreting it as an artificial chemistry, we are able to study how mutation and inheritance propagate through generations of bacteria.

This paper formally describes the bacteria model we used. We describe how bacteria are encoded as DNA, interpreted as artificial chemistries and executed with environmental input. Initial experiments with colour inheritance are explored, setting the stage for experimentation in bacterial evolution, where DNA, gene regulatory networks, and observable behaviours are evolved.

2 The Virtual Bacteria Culture

We have developed a simulation tool that models each bacterium as an agent on an agar plate, which is implemented as a 2-dimensional grid (Figure 1). Bacteria, residing on the cells, are able to receive environmental input and generate emissions. As an artificial chemistry it provides an operating system where bacterial behaviour programs, modelled through gene regulatory networks, are generated from a DNA-like encoding and executed, thus continually updating the bacteria’s states. The artificial chemistry uses explicit rules to simulate a virtual physical world, where electrical charges form the basis of interactions among proteins within a bacterial cell.

The strategy of a discrete two-dimensional environment with agents moving in continuous space was adopted due to its success in other simulations [13, 14, 15]. This strategy allows agents to assess their locality within the world quickly, hence, facilitating local interactions.

The discrete 2D environment is implemented as a cellular automaton (CA), where each cell keeps track of the concentrations of various nutrients and toxins. However, diffusion and decay of nutrients propagate in a continuous manner across the cellular grid. The explicit locality in CA is essential since massively parallel local actions dictate the global behaviour of the system. Our bacteria move, consume nutrients, hunt, and reproduce, so assessing their location quickly is an essential attribute of the model.

By using a 2D automata upon which a swarm of bacteria move, we can simulate each agent with great detail without a corresponding increase in environment complexity.

3 Bacterial Gene Regulation Network and Metabolism

Following the artificial chemistry (AC) approach proposed in [3], we define a bacterial gene regulation network and its metabolism as a triplet \( \Phi = (S, R, A) \). Here \( S \) represents the set of elements in the chemistry, \( R \) is a set of interaction rules among the elements, and \( A \) is an algorithm that uses the rules in \( R \) to update the concentrations of the elements in \( S \).

3.1 Elements \( S_n \)

Each bacterium \( n \) in the culture maintains and evolves its own chemistry \( \Phi_n = \{S_n, R_n, A_n\} \). The set of elements \( S_n = (G_n, P_n, C_n, B_n) \) is specific to a bacterium:

- \( G_n \) is the set of all genes in the bacterial genome, with promoter and inhibitor regions (Section 3.4).
- \( P_n \) is the set of all basic proteins produced by the bacterium.
- \( C_n \) is the set of all bounded complexes of proteins and proteins bound with promoter and inhibition regions of genes.
- \( B_n \) is the set of observable behaviours (external actions) the bacterium can execute.

Each bacterium hosts its own genome \( G_n \), from which it generates its characteristic proteome—that is, its basic proteins \( (P_n) \), complexes of proteins \( (C_n) \), and their interaction rules \( (R_n) \). The reaction rules \( R_n \) describe both how assemblies of proteins are composed and how proteins, or combinations of proteins, trigger behaviours \( (B_n) \) of the bacterial cell (Section 3.5).

Hence, a bacterial artificial chemistry \( \Phi_n \) defines a state machine within a bacterium, where cell-internal reactions among the proteins and complexes occur along with their changing concentrations, eventually resulting in observable behaviour of the bacterial cell, such as tumbling or replication by division. In fact, the bacterial chemistry emerges from the gene regulatory network defined by genome-site binding, gene activation, and gene inhibition rules of proteins (see Section 3.3 and [11] for more details on our implemented model of chemotaxis in \( E. coli \)).

3.2 Proteins \( P_n \) and Protein Complexes \( C_n \)

Proteins are formed as folded chains of amino acids. For the following descriptions we use a one-letter encoding for the 20 amino acids found in nature (Table 1). Formally we represent proteins \( p \in P \) as non-empty words over the amino acid alphabet \( \Sigma = \{A, C, D, ..., W, Y\} \), such that \( P \subset \Sigma^+ \).

According to the interaction rules \( R_n \), proteins can bind with other proteins to form larger complexes. These recursive compositions of proteins into complexes can be defined as follows. We recursively define sets of complexes as words over the extended alphabet \( \Sigma' = \Sigma \cup \{'\sim\}' \). Starting with the set \( P \) of basic proteins,

\[
C^0 = P_n,
\]

we compose level-1 complexes by appending a basic protein to a complex of level \( i - 1 \):

\[
C^i = \{c \sim p \in \Sigma' \mid c \in C^{i-1} \land p \in C^0\}.
\]

The (infinite) set of all possible complexes,

\[
C = \bigcup_{i=0}^{\infty} C^i,
\]
Table 1: Amino Acid Codings and Simulated Charges.

<table>
<thead>
<tr>
<th></th>
<th>Charge</th>
<th>Amino Acid</th>
<th></th>
<th>Charge</th>
<th>Amino Acid</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1.81</td>
<td>Alanine</td>
<td>G</td>
<td>8.5</td>
<td>Glycine</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>1.28</td>
<td>Cysteine</td>
<td>H</td>
<td>-8.3</td>
<td>Histidine</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>-6.64</td>
<td>Aspartic Acid</td>
<td>I</td>
<td>7.0</td>
<td>Isoleucine</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>-3.5</td>
<td>Glutamic Acid</td>
<td>K</td>
<td>-2.6</td>
<td>Lysine</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>3.67</td>
<td>Phenylalanine</td>
<td>L</td>
<td>-3.6</td>
<td>Leucine</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>-5.7</td>
<td>Methionine</td>
<td>S</td>
<td>-0.5</td>
<td>Serine</td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>3.3</td>
<td>Asparagine</td>
<td>T</td>
<td>1.0</td>
<td>Threonine</td>
<td></td>
</tr>
<tr>
<td>P</td>
<td>1.0</td>
<td>Proline</td>
<td>V</td>
<td>-6.4</td>
<td>Valine</td>
<td></td>
</tr>
<tr>
<td>Q</td>
<td>2.5</td>
<td>Glutamine</td>
<td>W</td>
<td>5.3</td>
<td>Tryptophan</td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>-3.1</td>
<td>Arginine</td>
<td>Y</td>
<td>-9.3</td>
<td>Tyrosine</td>
<td></td>
</tr>
</tbody>
</table>

Table is the superset for the (finite) complex set formed within a bacterium: \( C_n \subset C \).

3.3 Protein charges

The charges used in this implementation, listed in Table 1, were selected randomly. This table of charges dictates how the whole system will behave since the rules are generated solely on the basis of these amino charges. Since proteins are strings of these amino acids, their charges are simply the sum, namely

\[
\text{charge}(Pr) = \sum_{i=1}^{n} \text{charge}(\text{amino}_i).
\]

3.4 Genes \( G_n \) and Regulation Networks

Gene regulation networks (GRN) determine how and when genes are turned on or off (activated or deactivated). Through the processes of transcription and translation, proteins are produced from active genes. Some of these so-called regulatory proteins (transcription factors) control the activation of other genes by promotion and/or inhibition. Through the interaction among active genes, proteins, and regulatory sites on the genome a network of signal transmissions is established (GRN). In addition to environmental inputs, also signalled to the cell with the help of proteins, the GRN controls when, where, and how much protein is produced at each point in a cell’s life span. In the bacterium Escherichia coli (E. coli), for example, these networks determine the assembly of the flagellar apparatus and its chemotaxis, which controls the bacterium’s movement [9, 8].

Bacterial genes are encoded as modules, called operons, which are subdivided into two key regions, usually adjacent or relatively close to each other on the genome strand

\[\text{charge}(Pr) = \sum_{i=1}^{n} \text{charge}(\text{amino}_i).\]

Figure 2: (a) An operon-like gene structure consists of two regulatory regions (promoter and inhibitor site) and the gene-product region encoding for a protein. (b) Promoter A activates the production of protein B. (c) Repressor C prevents protein A from binding, thus inhibiting production of the encoded protein.

[7]: (1) regulatory binding sites and (2) sites encoding for a single protein or several proteins (Fig. 2(a)). For example, when a repressor protein binds at an inhibitor site, it prevents RNA polymerase—an enzyme that serves as the reading apparatus during transcription—to bind at any nearby sites. Some of these bindings create a conformational change in the physical DNA structure, allowing RNA polymerase to transcribe the encoded protein [1]. In Figure 2(b) molecule A binds to the promoter site, triggering the production of the encoded protein B. However, this promotional effect ceases when a repressor protein C is bound to an inhibitor site, where it blocks RNA polymerase from initiating the transcription process, which consequently deactivates the gene, and stops the production of the encoded protein (Fig. 2(c)).

Following this simple operon model, our virtual bacteria genomes consist of sequences of genes \( g \in G_n \) (operons) with a promoter region \( R_{pro} \in C_n \), an inhibiting region \( R_{inh} \in C_n \) and a codon sequence, \( R_{enc} \in P_n \), that encodes for a single protein. Formally, we represent all operons \( G_n \) as words over the extended alphabet \( \Sigma' = \Sigma' \cup \{'-\}' \):

\[ G_n = \{R_{pro} - R_{inh} - R_{enc} \in \Sigma'\}. \]

For example, the gene \( MM-AB-MYYA \in G_n \) encodes the protein \( MYYA \) and has promoter region \( MM \) and inhibitor region \( AB \).
3.5 Bacterial Behaviours

Though the interaction of proteins can generate gene regulation networks, it is the observable life functions that can be seen directly. The goal of the entire gene regulation strategy is to eventually form complexes that adhere to user defined patterns which, in turn, invoke specified functions (Table 2).

<table>
<thead>
<tr>
<th>Global Rule</th>
<th>Bacterium Elements</th>
<th>New Rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>•R!Q! → split()</td>
<td>•T~MRMAQTHPYA &lt;br&gt; • MKDYNGP-TQRTHPMLI-ACNHV &lt;br&gt; •MRMAQTHPYA</td>
<td>•T~MRMAQTHPYA → split() &lt;br&gt; • MRMAQTHPYA → split()</td>
</tr>
</tbody>
</table>

Table 2: Two elements from a bacterium’s artificial chemistry match global pattern R!Q!, resulting in new rules which can trigger splitting. The ‘!’ represents any string of arbitrary length over Σ.

The regulatory networks thus control the frequency and timing of these functions, with environmental feedback influencing fitness. In this implementation, we have defined the following nine functions:

- **Split()** causes the bacterium to split into two new bacteria at the same position. The DNA is mutated before the two new bacteria are created.
- **Die()** causes the bacterium to die, outputting it’s DNA and life statistics.
- **Tumble()** causes the bacterium to reorient itself randomly in 360°.
- **Move()** steps the bacteria along its direction vector according to its speed.
- **ExpelToxin()** expels toxin at a pre-defined rate
- **AddEnergy()** adds energy to the bacterium’s total energy supply. Energy is required to move and to reproduce.
- **AddToxin()** adds toxin to the bacterium from the environment.
- **BeHungry()** sets a boolean to indicate that this bacterium can eat other bacteria. Eaten bacteria are digested and their molecules are incorporated into the bacterium’s chemistry.
- **SexualReproduction()** sets a boolean to indicate that this bacterium can reproduce sexually (crossover mutation), and will produce offspring at the first possible instant with a mate at a neighbour site.

Though these functions by no means define all life processes in a bacterium, they are a simple test bed for the concepts being implemented in this paper. More functions could easily be added.

3.6 Rules

The motivation for developing an artificial chemistry as the basis of the bacteria was to achieve a state machine or operating system for a bacterium’s life. The rule generation is independent from the AC, since there only the rules and the concentrations of proteins and complexes matter. By generating rule sets from DNA, any DNA can be tried and the resulting AC executed as the bacterium lives and acts in its environment. The set $R_n$ of rules is generated for each bacterium $n$ depending on the elements the bacterium possesses. Each rule has three parts:

- a probability of executing each step
- a set of input elements in $S_n$, and
- a set of output elements and behaviours in $S_n$.

If all the input elements are present when a rule executes, the concentrations are updated. Input elements (left-hand side of a rule) are removed from the reservoir of proteins and complexes, whereas output elements (right-hand side of a rule) are added. When an output element corresponds to an observable behaviour, that function is executed immediately. Figure 3 illustrates a bacterium executing a single rule, updating the concentrations of elements involved. This rule-based approach permits the generation of rules that model protein binding, gene activation and gene inhibition, as has been shown for the specific case of *E. coli* chemotaxis [11].

![Figure 3: A single rule execution where concentrations are updated from iteration step $t$ to $t + 1$.](image-url)
4 Genome Interpretation

We model the interpretation of DNA as the generation of a bacterium’s artificial chemistry. By transcribing the DNA, the basic proteins and genes in the AC can be decoded and used in the creation of the rules. Global rules regarding behaviour define the desired patterns and therefore determine how a bacterium will be interpreted as an AC.

The gene sequences are modelled as strings, with no physical properties relating to structure. Because we use a DNA-inspired 4-letter alphabet \( L = \{ A, C, G, T \} \), mutation operations on the bacterial genome are easily implemented in a genetic algorithm. The random selection of bases for point mutations can modify proteins produced by a bacteria and change the very way the gene is transcribed, for example by mutating a stop codon.

The 64 possible codons—all 3-letter words over \( L \)—are mapped onto the 20 amino acids, following the encoding scheme on natural DNA [1]. This redundancy gives the DNA a robustness from which stability can arise despite mutation. Single character amino codes simplify the description of the proteins, and are thus used throughout the simulation.

4.1 Gene Transcription

The generation of a bacterial phenotype from a genotype begins with genome transcription. The algorithm is loosely based on gene transcription in E. coli. Transcription defines a new protein wherever it encounters the start codon “ATG” (M). It continues reading codons until it encounters one of the stop codons “TAA”, “TGA” or “TAG” (*).

Inhibitor regions for each gene are identified by finding the first occurrence before the start codon of “YN” and setting the inhibitor region to equal the sequence from after “YN” up to “M”. Similarly, promoter regions are calculated by finding the first occurrence of “LT” and setting the promoter region equal to the sequence from after “LN” to “YN” (Figure 4).

```
LT GLARTRA YN TVPVVRI | M RMAQTHPYA *
```

Figure 4: Sample gene with delimiting regions in bold.

4.2 Rule Generation

After the basic genes and proteins have been defined, rules are applied to find out which proteins could bind, inhibit, or promote other molecules in the set \( S \). There are four types of rules which are generated, all using the amino acid charges defined in Table 2:

- **Complex Binding**: Two or more proteins or protein complexes combine to form a larger protein complex refsec:Proteins.
- **Gene Binding**: Proteins or protein complexes bind to inhibitor or promoter sites on a gene (Figure 2).
- **Gene Activation**: Genes with proteins bound to their promoter regions produce the encoded protein (Figure 2(b)).
- **Gene Deactivation**: A protein binding on an inhibitor site terminates the expression of the encoded protein (Figure 2(c)).
- **Behaviour Generation**: Proteins matching global patterns enable behaviour (Table 2).

5 Emergent Bacteria Patterns through Colour Inheritance

One goal of our system was to provide a visualization of bacteria as they evolve. Each time a bacterium splits into two, the RGB components of the colour are each mutated within a radius. By limiting inheritance to colour, emitted patterns emerge which exhibit an element of self organization. Some patterns were predictable while others seemed creative, like art. This section provides examples of bacteria growth patterns and evolutionary colour inheritance in bacteria.

5.1 Bacteria Circle

When a single bacterium is placed into agar and allowed to grow and multiply, a circle emerges as food is depleted in the centre and bacteria move outward. A more interesting pattern emerges when symmetry is broken and each bacterium is restricted to mutating only one colour as it evolves (Figure 5).

![Figure 5: Inheritance Patterns.](image)

This culture begins with 100 bacteria in the centre and a uniform food distribution. After 10 hours, the culture has reached the edge of the square plate with a population peak
of 997,631 bacteria. There are distinct groupings of dark and light coloured segments as well as combined regions.

Some segments of the population exist within large pockets of similar individuals while others mix more thoroughly with non-similar bacteria. This visualization illustrates how locality and inheritance are related. Furthermore, it shows the non-uniformity of a bacteria environment, an important fitness consideration for evolution.

### 5.2 Creative Bacteria Patterns

When bacteria are seeded in various places with different initial colours, speeds and mutation radii, more complex and less predictable patterns emerge. Some of these patterns seem almost creative—a product of evolution and self organization.

Figure 6 shows a culture at nine time steps during its evolution. The first image shows many small, independent cultures distributed throughout the environment. The second and third images show a smoothing of these emissions into larger regions of colour with bright boundaries inbetween. The second row of images illustrates how the movement by bacteria and the decay of the colours begin to form seemingly coordinated patterns. Twelve hours later, the third row shows the bacteria have continued to emit colour, and more detailed and unexpected patterns have emerged.

### 6 Experiment in Bacterial Evolution

Having built the syntax for describing bacteria and experimented with simple colour inheritance, we explored the evolution of bacteria genes and traits through inheritance. Without explicit fitness, our simulation was free to explore
for creative solutions. A single random set of global patterns (Table 2) was used for all behaviours explored in this paper. A string of DNA, with 4 random genes was created as an initial genotype for the first bacteria in our culture (Table 3). For these experiments the mutation rate was set at 0.01, meaning each base (letter from L) has a 0.01 probability of being mutated. Only point mutation and standard single-point crossover were studied in this paper.

### 6.1 Genome Mutation

Since a bacterium’s genome is a string of a 4-letter alphabet (A,C,G,T), mutation simply mutates any letter into any other with equal probability. Because transcription is defined so rigorously, mutations in the genome could result in either large or small changes.

Crossover occurs when two sexual bacteria come in close contact to produce an offspring. Crossover defines a random point within the size of the smaller genome and recombines the two parents genetic data together into a new bacteria.

### 6.2 Evolution Example

Table 3 lists the DNA and metrics of the associated artificial chemistry. Notice how, for example, from parent to child A there was a tremendous simplification of the AC from 604 rules to 27, yet the child still retains 4 genes and only lost the AddToxin behaviour. Child B has lost an entire gene, yet has an even simpler chemistry which contains all the same behaviours as child A. The ability of evolved organisms to simplify as well as increase their complexity is important for avoiding successively more complex bacteria. Child C demonstrates how new behaviours can be acquired, even as the number of genes decreases to two.

This example illustrates how small changes in DNA (from parent to child A) can have a large effect on the phenotype of the organism. It also illustrates the robustness of gene encoding, since all four bacteria exhibit similar behaviours.

### 6.3 Other Results

We have performed other tests with additional interesting results. Among the most interesting behaviours that evolved in other experiments were the tumbling frequencies. As the proteins change, the probability of tumbling changes, resulting in tumbling frequencies from 0 to 10 times per second. A variety of swimming motions were observed. Some swarm without tumbling, others tumbled frequently and appeared to jitter very quickly. Still some others tumbled so much they did not end up moving much at all.
Hunger was another interesting behaviour because of its emergent stability. At first, hungry bacteria would eat their own offspring, gaining their energy. However as this precluded the continuation of the species, only fast swimming or lucky bacteria were able to survive. As the density of the bacteria lowered, the number of hungry bacteria stabilized.

7 Conclusions

In this work we introduced a model and a software system that simulates the evolution of bacteria. Transcription algorithms allow bacteria to be encoded as DNA then interpreted as an artificial chemistry. These artificial chemistries then interact with the environment, taking in nutrient and emitting toxins. By relating the DNA phenotypes to AC phenotypes we were able to explore the evolutionary mechanics of bacteria cultures. Expected emergent patterns such as the bacteria circle (Figure 5.1), affirm the validity of the implemented agent behaviour.

When exploring bacteria evolution, many interesting observations can be made.

- A single mutation can result in either no change, a single codon change, or affect how the gene is transcribed.
- Environment and inheritance determine a bacterium’s chance of survival (implicit fitness).
- Bacteria patterns exhibit creativity and organization. Through the development of a bacteria model with environmental interaction, the analog processes in nature become more clear. The self organization observable everywhere in biology is imposed by the underlying physics controlling the system. Global rules guide evolution through constant interaction with the environment and each other. Order is a result of interacting molecules, balancing charges and complementing shapes. This environmental stress not only “evaluates” individuals for fitness but impacts the very interactions that guide an individual’s behaviour. Since genes can be turned “on” or “off” by mutating start or stop codons, genes can become dormant until they are “reactivated”. Modelling bacteria at this level has been successful as many observable behaviours were seen and interesting behaviours evolved.

Bibliography


[17] University of Wisconsin-Madison Department of Bacteriology, http://www.bact.wisc.edu/Bact303/
