Potential Adaptations to Artificial Chemistry

Adly Templeton

May 25, 2016

Contents

1 Abstract **3**

2 Introduction **4**

3 Background **6**
3.1 Hutton's System 6
3.1.1 Structure 6
3.1.2 Genetics 7
3.1.3 Optimizations 7
3.1.4 Physics 8
3.1.5 Selective Pressure 9
3.1.6 Other Work 9
3.1.7 Computational structure 12

4 The Artificial Chemistry Environment **15**
4.0.1 Atoms and reactions 15
4.0.2 Simulation details 17
4.1 A basic organism 18
4.2 The lifecycle of the organism 18
4.3 Membrane Behavior 21
4.3.1 Summary 24

5 Our Modifications **25**
5.1 Reaction Data 25
5.1.1 Hutton's enzymes 25
5.1.2 Advantages 26
5.2 3-way mutations 27
5.3 Basic Physical changes 28
5.4 Membrane Physics 29
5.5 Enzyme Physics 30

5.6 Degradation 31
5.6.1 Summary 32

6 Mutation **33**
6.1 Problems 33
6.2 Languages 34
6.3 Other Languages 36
6.3.1 Gliders 36
6.3.2 State transitions 37
6.3.3 Our mutations 39
6.3.4 Summary 39

7 Experiments and Results **41**
7.1 Basic Procedure 41
7.1.1 Data Collected 42
7.2 Experiments 42
7.2.1 Initial Failures 42
7.2.2 Basic Simulations 43
7.2.3 Lowering Mutation Rates 45
7.2.4 Declining Biomass 48

8 Conclusion **51**
8.1 Future Work 51

Chapter 1

Abstract

In an attempt to simulate rich evolutionary growth, Hutton has previously devised a system which simulates basic forms of artificial life in an artifical chemistry environment [8]. However, while Hutton's system demonstrated potential in several key areas, it failed to produce evolutionary growth. The successful development of a system with rich evolutionary growth would provide immense insight into evolutionary progress and mechanisms. We made several modifications to Hutton's system with the goal of stimulating evolutionary progress. These modifications included changes to the basic physics of the simulation, a new mechanism for mutation and new mechanisms for the storage and reproduction of enzymes. As in Hutton's work, we did not successfully create evolutionary growth or generate adaptive mutations. However, a key novel contribution of our current work is success in spontaneously generating interesting nonadaptive mutations, a result which Hutton's simulations did not attain. More work is required to produce a system that displays evolutionary growth, but our work demonstrates several features which might be useful for such a system.

Chapter 2

Introduction

Evolution arises from complex interactions between organisms. The creation of a virtual environment that demonstrates open-ended evolution would allow us to model evolutionary behavior. A potential medium for this environment are *artificial chemistries*, which simulate organisms at a low biochemical level, allowing interactions and competition between organisms [3, 8]. Here, we pose several possible modifications to artificial chemistries, particularly that of Hutton [8], to attempt to encourage richer evolutionary growth.

Hutton's artificial chemistry [8] can support complex organisms. This chemistry consists of atoms of various types and states. Different reaction rules can make and break bonds, or change atoms' states. Organisms consist of a membrane and a strand of DNA (although other designs have been proposed [9]). Genes on this strand code for a single enzyme, which catalyzes a specific reaction. This system can support very complex organisms: for instance, this system can construct gliders capable of universal computation [2]. However, complexity is somewhat limited by computational resources—complex organisms are extremely large (this implies that more efficient computation—such as a HashLife-esque mechanism [5, 12]— significantly increases the usefulness of the system). In addition, because reactions are hard-coded into the system rule set, not mutable features, the cell has little power to drastically change.

A modification to this system known as *enzyme artificial chemistry* [7] allows each atom to carry *reaction data* that acts as an enzyme for a specific reaction and can be copied directly. Hutton describes an organism consisting of just a strand of DNA without a membrane that exists without any external rules. We would like to bring together the complexity of Hutton's membrane-bound organism with the self contained nature of enzyme artificial chemistry.

We propose the creation of organisms within a modified version of Hutton's enzyme artificial chemistry. First, we allow reaction data to be stored as an additional feature of an atom, and for reaction data to be directly copied. This already signifi-

cantly reduces the size needed for a self-sufficient organism.

We also can increase the number of reactions stored per atom. This sharply cuts the number of genes needed and, therefore, the computation required. This change also opens up possibilities for horizontal gene transfer between organisms. A single atom, analogous to a plasmid of bacteria, which contains all enzymes needed for a behavior, can be passed as a unit between organisms. Similarly, we can construct malicious viruses and other organisms of this sort.

Mutation is fundamental to evolution, and the mechanism of mutation needs to be both powerful and directed. The new genetic mechanisms allow us to tightly control mutation when reaction data is copied. The exact mechanism of mutation is well suited to experimentation, but there are some factors to consider. Each field of the reaction can have a specific chance of mutation. Some prediction of useful reactions may also be helpful: for example, the value for post-reaction states can frequently mutate to the pre-reaction state. Another potentially useful factor is the ability to inactivate a reaction: this allows for increased genetic variability through silent mutations, similar to unexpressed biological genes.

With these modifications, we hope to create a vibrant virtual ecosystem. Such a system would many sorts of competition, information transfer, predation, and parasitism. Combining these elements together will create a system that models, with some complexity, the interactions of real organisms.

Chapter 3

Background

3.1 Hutton's System

Hutton [8] has constructed an artificial chemistry which can support complex organisms. Hutton's chemistry describes atoms in a two dimensional space. Each atom consists of a fixed type $\in$ {a, b, c, d, e, f} and a variable state $\in$ {0, 1, 2 ...}. Atoms may be bonded to some number of other atoms, which will remain nearby (within a Moore neighborhood of radius 2). Bonds may not be crossed at any time. Reactions are defined by rules that accept input atoms, of specific types and states, either bonded or nonbonded, and may modify both the states of the atoms and the existence of a bond. They may not modify the type of the atom. The symbols x and y represent variable states for these reaction rules [8]. Note that no such wildcard values can represent states. This has profound implications—for instance, this restriction makes the construction of a universal decomposition enzyme impossible.

3.1.1 Structure

Hutton's organisms consist of a bonded membrane of type a atoms. This membrane design allows nonbonded atoms to pass freely, but constrains bonded molecules, which cannot pass through the membrane without crossing bonds. A pair of three-atom reactions ensure that the membrane is the appropriate size: free type-a atoms are continually added to or removed from the membrane. As the nonbonded atoms drift away, the membrane tends towards the smallest size that contains the interior structures. The membrane's primary function is to ensure that only a particular organism can derive a benefit from the enzymes it produces. Without this behavior, short self replicators, which reproduce assisted by the free-floating enzymes produced by other atoms, dominate [7]. This parasitism lowers the benefits in fitness gained from new adaptations, decreasing evolutionary activity.

3.1.2 Genetics

Attached to the inside of this membrane is a strand of DNA, which we will loosely call *DNA*. This strand begins with a type e atom and ends with a type f atom. Genes consist of a series of types $\in \{a,\ b,\ c,\ d\}$, delineated by f atoms. Each gene codes for an enzyme which can catalyze a single reaction. This coding is represented numerically: the sequence of atomic types in a gene is translated into a single number, which becomes the state of the enzyme. A modular equation maps this numeric state to the reaction specified. Enzymes move within the cell as diatomic molecules, which prevents them from passing through membranes due to the constraints discussed above.

Hutton's organisms reproduce in a three-stage lifecycle. First, the cell creates a copy of its DNA parallel to the original strand. Next, the genetic strand splits, and the membrane, passed through the two strands of DNA, divides and pinches off. Finally, enzymes are created from each gene, restoring the DNA to its original state, and the process begins again. These three stages are constantly repeated—not controlled by the regulatory mechanisms in living cells. Reproduction is primarily limited by the abundance of *food* (free floating atoms in state 0). When reaction data is copied, there is a small chance of a random mutation, which provides the variability needed for evolutionary growth.

3.1.3 Optimizations

This system, in theory, can support very complex organisms. However, in practice, complexity is somewhat limited by available computational resources. The unrealistic computational costs required to encode every reaction into enzymes means that most reactions must be hardcoded into the system. As reactions are hardcoded into the system, and not mutable qualities, the cell has limited power to direct its own evolution. This hardcoding also creates a strong selection bias against complexity. Note that a long strand of bases is needed to encode a single enzyme. As the time and resources needed for reproduction is almost directly proportional to the length of the DNA, complexity is heavily selected against. The addition of an additional enzyme to a basic cell (which contains one gene in Hutton's system) results in a 100% or more increase in reproduction time. If all reactions were encoded as enzymes. In contrast, a new enzyme will only result in a marginal increase in reproduction time porportional to the increase in the size of the entire genome. While computational resources are far from the only limiting factor, they present many issues which must be addressed.

Hutton has also presented an adaptation of this system known as *enzyme artificial chemistry* [7]. In this system, each atom, in addition to its state and type, also carries

reaction data. That is, every atom can act as an enzyme for a specific (possibly null) reaction. A flag on each reaction determines whether the reaction data is copied over from one atom to another (as occurs during replication of DNA). In this chemistry, simple forms of life can exist without any external rules: Hutton describes a self-sufficient organism which consists of a strand of DNA without a membrane [7]. We will seek to adapt concepts from enzyme artificial chemistry into more complex organisms.

A third paper by Hutton provides another potentially useful element. An adaptation of this artificial chemistry, the *Organic Builder*, contains a special atom known as a *caustic agent*. Upon collision with an atom, the caustic agent will break all the bonds of the affected atom and reset its state to 0. Only type-a atoms (such as membranes) are immune to these effects [9]. This caustic agent was originally devised to incentivize enclosure with a membrane. However, the addition of caustic agents could have several beneficial effects for general evolution. The first effect is a reduction of the fitness of potential simple non-membrane-bound organisms, which might otherwise out-compete more complex life. In an enzyme artificial chemistry, a trivial one-atom 'organism' has the ability to reproduce very quickly—a situation obviously impossible in biological life. In the same way, caustic agents will help to reduce, though not eliminate, the effect of various secreted enzymes (such as hostile toxins). Such effects mimic the environmental decay inherent in biological life. Another, more important effect is the automatic decay of organisms after the membrane is split, especially by a hostile enzyme. Note that the structure of the reaction rules prevents a general-purpose decomposition enzyme. This gap requires some sort of universal decomposer, such as a caustic agent, to enable full predation.

3.1.4 Physics

Hutton described two possible implementations of a physics engine. The first is a discrete lattice in which atoms randomly move to nearby cells, restricted by bond crossing and stretching (we will now use the word 'cell' to refer to a unit of a grid and the word 'organism' to refer to a creature). The second implementation is a continuous physics system, in which atoms have positions and velocity, and forces (such as the stretching of bonds) act on the atoms. Hutton claims that both these implementations are functionally equivalent, though the discrete implementation can be expected to be faster [8]. However, other attempts to replicate a continuous physics engine faced severe problems, both in excessive computation times and unreasonable computational complexity and in strange and undesirable behaviors (the cell would occasionally invert, pushing out the DNA) [4].

These organisms, in a medium containing free-floating food particles, quickly grow exponentially, devouring all available resources or space. To allow for continued growth, Hutton introduced *floods*, which occasionally wipe out a quadrant

of the world and restore all atoms to food. This can be seen as a computational optimization to avoid simulating constantly-growing regions of space. However, this optimization carries some potential disadvantages: mutations formed in the corner of the quadrants have little chance to propagate farther (large molecules, such as organisms, move little to no net distance). For this reason, much simulation time is wasted evolving 'doomed' cell lines, a problem which grows with the size of the grid.

3.1.5 Selective Pressure

As the time required to reproduce is directly proportional to the size of the DNA, and as the original organisms do not code for any useful enzymes, natural selection tends to favor those organisms with the shortest DNA, creating an inherent evolutionary degeneration. Hutton showed that, if one of the necessary reactions are removed from the hardcoded rule-table and instead seeded as an enzyme to some (but not all) initial organisms, complexity would be maintained despite the trend towards degradation. In a simple evolutionary experiment, a second enzyme, which allowed the organism to use a second type of food (free-floating atoms in state 36) was able to out-compete simpler organisms, showing the application of a basic principle of natural selection [8].

These two experiments demonstrate that Hutton's organisms contain the basic ingredients necessary for complex evolution—however, they have not displayed any satisfying evolution without heavy experimenter intervention. Hutton presents several explanations for the lack of interesting evolution. The first is the size of the mutational jumps: the appearance of a functional enzyme requires a precise and excessively long sequence of types. A sharp selection pressure, combined with the precise nature of the enzyme coding mechanism, prevents these mutations from appearing. Ideally, these organisms would contain all the enzymes needed to sustain their growth. However, the size of such an organism - at least a 500 atom long DNA strand—renders such an organism computationally impractical. A second concern is the lack of immediately beneficial adaptations—relatively simple mutations that will convey a direct benefit to the initially simple organisms [8]. This problem is likely significantly more central and harder to fix.

3.1.6 Other Work

"What characteristics are necessary to support creative open-ended evolution" is a complex unanswered problem. Taylor presents an informative discussion of this topic [13]. First, Taylor argues, we must reach a suitable definition of "creative open-ended evolution" Many artificial life systems, such as genetic algorithms, can show progress towards a definitive goal, even progress not envisioned by the researcher. However, the progress of genetic algorithms generally stops upon reaching its peak.

Continual progress can be seen in other systems, such as Ray's Tierra system, but these adaptations do not produce any fundamentally new results, and are still not satisfying evolution. These systems produce a repetitive cycle of quantitative shifts but no net change. Taylor presents some loose definitions of evolutionary creativity, but a rigorous and precise definition remains elusive [13].

The first, and least controversial, requirement presented by Taylor is a lack of a *fixed fitness function*. A fixed fitness function is a static function which assigns a numerical fitness value to each organism, as is used in genetic algorithms. A system in which the fitness of an organism is constant will find only fixed local maxima, which will not allow creative evolution. Instead, we seek to find creativity in the ecological and physical interactions between organisms, in such a way that the fitness of an organism is intrinsically linked to the organisms around it. As an extension of the first requirement, Taylor proposes that organisms must be "embedded in the arena of competition" [13]. This design allows for arbitrarily complex interactions between organisms in ways not specified by the researcher [13]. Artificial chemistries (as opposed to machine-code-based artificial life) are inherently embedded, and satisfy this requirement.

Taylor also proposed additional specifications. First of all, Taylor argues, the basic self-replication mechanisms could be embedded as a feature of the environment. Such a situation would be equivalent to Hutton's original organisms, which reproduce primarily using a universal hardcoded rule table. Any DNA, then, would produce phenotypes not necessarily related to reproduction [13]. This change, however, requires that suitable initial phenotypic improvements exists as the "first rung of the ladder"—the same problem experienced by Hutton's original organisms would be made more drastic. Instead, some degree of initial complexity, directed towards self-reproduction, could provide a scaffolding off which more useful adaptations could be based.

Taylor also argues for the necessity of *materiality*. A system is material if it has some notion of conservation of mass. Artificial chemistries, by definition, satisfy this requirement (other cellular automata, such as the Game of Life, do not). Materiality enables predator-prey relationships, as every organism becomes a resource for every other organism.

The necessity of energetics, however, is more contested [13]. Dorin provides one example of an artificial chemistry that supports energetics [3]. In this system, energy is a property of a linked molecule (equivalent to an organism). Incident sunlight increases the energy of all atoms during the daytime, and free energy dissipates naturally. Energy is also stored in bonds, and high-energy bonds, the equivalent of *sugars*, store energy for later use. The creation of a bond takes energy from the attached molecule, which the destruction of a bond releases.

Dorin's artificial organisms are incomplete—they contain no capacity for growth,

replication, or mutation. They only serve as an proof of concept for the ecosystems which can be formed from a relatively simple energetics model. Dorin's energetics provides varied ecological niches, a property crucial to rich evolution. Dorin outlines potential photosynthetic autotrophs, chemosynthetic autotrophs, decomposers and heterotrophs, all of which occupy their own ecological niche [3]

However, problems arise when translating this model of energetics to complex organisms, such as Hutton's organisms. Hutton's organisms, at least initially, have no regulatory mechanisms to control the release of energy (nor could one be simply implemented). For instance, in order to add an additional base of DNA, the appropriate atom must collide with the appropriate base at the same time as the appropriate sugar molecule collides with the appropriate decomposition enzyme (or close enough temporally so that the energy does not dissipate). Such a system would drastically slow down the simulation without adding much, if any, evolutionary richness. A rich and practical system of energetics remains elusive.

Earlier forms of artificial life, such as the Avidia system and its precursors, envision organisms as machine code programs. The Avidia system consists of 'living' programs running on a simulated CPU. Given limited resources, these programs must reproduce. This approach has several advantages. In particular, most pieces of an organism's genetic code (which translate to instructions in a program) have a direct and significant effect on the phenotype. Organisms which perform mathematical tasks from the environment correctly are able to increase their *merit*, which increases the CPU time available for reproduction. These calculations provide appropriate early targets for evolution. This system shows significant evolutionary activity, especially towards the solution of presented mathematical problems [10]. However, this system does not display rich interactions between organisms. Though interactions such as parasitism can exist, they rely on hardcoded commands. Therefore, organisms cannot evolve fundamentally new or interesting behaviors, and are limited to interactions provided by the researchers.

Forms of artificial life based on cellular automata or artificial chemistries, such as Hutton's organisms, take a radically different approach to artificial life. Machine code systems, such as Avidia, can easily harness powerful phenotypic effects. Code-based life allows organisms to use complex decision-making schemes. One initial concern with cellular automata-based life is that these complex behaviors may not be possible. However, cellular automata can harness the same computational abilities. Cellular automata provide such a powerful framework that some even suggest that cellular-automata based software is useful for real world computations [1]. For instance, a basic two-state two-dimensional cellular automata, *Rule 110*, is known to be Turing-complete [2]. In this simple automata, activated cells will propagate leftwards and die from overcrowding. This creates various *gliders*, particles which propagate through the system. Different types of gliders move at different speeds and, upon collision

with another glider, produce some resulting gliders. A carefully-arranged initial set of gliders can be used to produce a *cyclic tag system*, which is equivalent to a Turing machine [2].

Though the size of such systems rules out a direct simulation of a Turing machine through rule 110 in our artificial chemistry, similar constructions of gliders can be used to give organisms complex behaviors. Several examples of propagating 'gliders' already exist in Hutton's organisms. The most prominent of these is the chain reaction which, moving through the DNA, splits the strands. However, a much more involved practical example, which shows the ability of nonadjacent atoms to communicate effectively, has been demonstrated. In an alternative method of DNA replication, a glider passes through the DNA. This glider encodes the type of the next atom to be added. When the atom is added, the process begins again [9]. Organisms in Hutton's artificial chemistry could use similar systems to transmit information or make complex decisions with power equivalent to that of machine-code systems.

3.1.7 Computational structure

In many ways, the limiting factor of an artificial chemistry system is the speed of computation. The most direct effect is that a sufficiently powerful system of evolution will produce novel results at a rate proportional to the speed of the computation. However, other, more potent improvements result from various trade-offs used in the design of artificial life. For instance, if the available computing power were to be increased by several orders of magnitude, it would be possible to simulate versions of Hutton's organisms which would encode enzymes for every reaction.

To that end, we seek any changes which would significantly increase the speed of the computation. One promising algorithm, *HashLife*, has been used to significantly speed up simulations of Conway's game of life. The HashLife algorithm represents the life universe as a tree, where nodes correspond to a square area and leaves correspond to a single cell. Each such square area of size n can concretely determine a concentric area of size n - 2 in the next generation (the edge cells depend on their neighbors, which are not contained within the square grid). The drastic performance improvements result from memoization and canonicalization of these higher-level nodes. We can then allow higher-level nodes to compute multiple generations at once, proportional to the size of the square area enclosed. This carries the caveat that the area guaranteed by an initial square area shrinks as the time computed increases—uncertainty advances from the edges at a rate of one cell per unit time. This algorithm leads to enormous increases in the speed of the computation of the Game of Life [5, 12]

However, several problems arise when transferring this algorithm to our artificial chemistry. The first is the nondeterministic nature of our artificial chemistry. Some

components, particularly the motion of atoms, are randomized. In other words, two sections of space with the same atoms will not necessarily produce the same result, breaking the assumption central to the HashLife algorithm. While the result of a sector might be given in a probabilistic table, this significantly complicates the algorithm, reducing any performance benefits. Other concerns relate to the lack of structure, especially recursive structure, in our artificial chemistry. Even in the Game of Life, HashLife only delivers significant improvements on patterns with high symmetry in our system. For several reasons, we do not expect to find high degrees of symmetry, especially symmetry aligned to square grids. The exact physical arrangement of the atoms is much more flexible than that of the Game of Life. One possibility is to abandon square regions and perform computations on the level of the organism. However, finding an efficient model of a cell with a given DNA is complex, error prone and computationally expensive, especially when probabilistic effects are taken into account.

A third obstacle is the *speed of light*, or the speed at which information and causality can travel within a system. This speed, which is one cell per unit time in the Game of Life, is equal to the rate of attenuation of the area determined by a square region n steps into the future. In our current system, the speed of light is equal to the enzyme range (a typical value of which might be 10 cells). This high speed means that a given region is able to determine little, if any, cells in isolation. However, though the maximum speed of light is high, almost all communications will occur at a speed of one cell per unit time or slower — the speed at which atoms move and chain reactions occur. For this reason, we can artificially lower this speed to 1 without any significant effects on the simulation.

If finding significant improvements to the computation speed is not possible, another useful ability is a multi-threaded simulation. Such a simulation could take full advantage of multi-core processors or even multiple physical machines. One such architecture is the *Movable Feast Machine*, designed to simulate cellular automata similar to ours—asynchronous, large-neighborhood cellular automata. This system, designed to perform real-world reprogrammable computations, allows multiple processors to operate on the same map. To allow this, a system of *inter-tile locking* freezes all nearby cells while a cell is being update. Each processor obtains some random unupdated cell, checking that it is not the neighbor of any cell currently owned by another processor, and performs the needed computations [1]. A second solution is to divide the universe into regions which are assigned to individual processors. This large-scale division reduces the cost of distributing individual cells to different processors. This carries the caveat that the boundary zones must be computed later by a single processor. Here, a much lower speed of light is very useful - it lowers the size of the locked regions or the border zones.

Consideration must also be given to the shape of the grid used. Hutton originally

placed the atoms on a square grid, but other grid shapes offer intriguing possibilities. A simple change would be the use of other regular polygons, such as hexagons, instead of squares. The primary effects of such a change results from differences in the number of cells connected to a given cell. A more involved change would be to use an irregular tiling, such as a Penrose tiling. An experiment investigating the Game of Life on a Penrose grid gives some insight into the effects we may expect. The most significant quantitative effect of a Penrose grid is a significant reduction in the *Lifetime to Stability*, the duration of chaotic activity generated by an initial configuration. However, this effect is counteracted by a significant qualitative increase in the frequency of complex large-period oscillators [6]. The effects of a different grid type on our artificial chemistries remains to be seen. However, any move to a non-square grid must be weighed against the increased complexity of computations.

Although our system has extremely rudimentary physics, implementation of selected high-level physical effects may prove beneficial. We are particularly interested in the implementation of *flow*. Persat provides an overview of the effects of fluid flow on physical bacteria [11]. Though most of the effects do not apply to our simple organisms, Persat details a number of useful results from fluid flow. In particular, fluid flow leads to *advection*, which carries solutes into and between cells. On small scales, this effect can outweigh diffusion as a means of transportation. Advection also increases the effectiveness of both nutrients and secreted compounds [11]. This boosts the success of the secreted compounds, either offensive or social in nature, that we would hope to see appear. However, we expect advection to be useful primarily in increasing the rate at which nutrients impact organisms. Without fluid flow, as atoms display no net movement on average, it is relatively rare that a nutrient atom makes contact with an organism. This scarcity is the primary limiting factor on the rate of reproduction. Consequently, increasing this rate significantly would result in many more divisions per cycle. This effect is analogous to an increase in computational efficiency discussed above—it is desirable because it allows more useful result with less computational time.

We have detailed the basis in the literature for our future system, especially the work of Hutton. We will discuss Hutton's system in more detail, as well as our own modifications, in Chapter 3.

Chapter 4

The Artificial Chemistry Environment

We will now discuss our basic artificial chemistry environment in greater detail. While our environment resembles that of Hutton, it diverges in a few key places. From this point on, we will refer only to our discrete-space implementation of Hutton's physics, ignoring a potential continuous-space interpretation.

It is important to remember that the physics of our system is constrained in two dimensions. That has important implications on the potential of our design.

4.0.1 Atoms and reactions

First, we will take a more formal approach to simple atoms. In our system, an *atom* consists of four attributes: A *type*, a *state*, some number of *bonds* and an optional *reaction data*. The type is an immutable feature of an atom represented by one of several letters (we choose the letters $A...F$). The state is a positive integer value which is mutable through reactions. This value may be unbounded, though we define a relatively large maximum value for performance reasons (we expect this to have little effect on the simulation). The primary function of both the type and the state is to store information and to determine eligibility for reactions. Individual values for type and state are largely arbitrary and only meaningful in the context of specific reactions (some exceptions to this will be discussed later). Atoms may also contain some number of bonds to other atoms. These bonds can not only be used to determine eligibility for a reaction, but also carry certain physical constraints on movement which will be discussed later.

The final attribute of an atom is the reaction data. This optional attribute is a fixed-length array of the reactions an enzyme may catalyze. The number of reactions per atom is a parameter of the simulation. Reaction data can catalyze appropriate

reactions within a given range of cells, defined by another parameter of the simulation (which we have set to 10 cells). However, a series of constraints apply to this catalytic action. Enzyme action is blocked by membranes to isolate the effect of inter-organism enzymes, both destructive and beneficial. This mechanism reduces parasitism and encourages evolutionary growth [8]. Formally, enzyme action is blocked if the line from the enzyme to either of the reactants crosses a bond between any two type-A atoms *and* if the enzymes and one of the reactants is not *connected* (joined by a chain of bonds). If either of the reactants are part of a membrane (formally, if they have type A and $30 < \text{state} < 40$), the enzyme must be connected to one of the reactants. These rules will have drastic effects on inter-organism interactions.

Reactions occur after two reactants move near each other. Hutton required that the two reactants be in each other's Moore neighborhood [1] [8]. We have found, however, that increasing this range to a Moore neighborhood of radius 2 allows for greater flexibility and quicker reproduction (this, however, incurs a performance penalty).

Reactions are defined by a one-line string. This string dictates the types, states, and bondedness of the reactants involved and the products formed. An additional boolean flag allows for the reaction data of the first atom to be copied to the second. For example, the string

$$A1 + B2 \text{ to } 3 - 4 \tag{4.1}$$

can be translated as "A type-A atom in state 1 interacts with a type-B atom in state 2 which are not bonded to the type A atom; the type A atom moves into state 3, while the type-B atom moves into state 4, and the two atoms form a bond". Two wildcards for the type values (but not for state values), *X and Y* are available. These wildcards allow reactions between atoms of any type. If the wildcards are the same, such as in the reaction

$$X5 + X0 \text{ to } 6 - 6(cpy) \tag{4.2}$$

the types of the reactants must be the same. (This is the reaction which adds atoms to a parallel DNA strand — the *(cpy)* flag copies over the reaction to the newly incorporated DNA atom).

We can also allow three way reactions. Both the justification and uses of these reactions will be discussed later, but it is helpful to describe the notation used.

$$A1 + B2 - C3 + \text{ to } 4 - 5 - 6+ \tag{4.3}$$

The trailing symbol indicates whether the third atom is bonded to the first, both before and after the reaction. Also note that reaction data can not be copied during

[1] A Moore neighborhood is the collection of cells which share one or more corners with a given cell

a three way reaction due to notational issues. We expect this restriction to have little effect.

4.0.2 Simulation details

During each iteration of the simulation, all atoms are moved in a random order. For each atom, the valid movement positions are determined according to a list of constraints. An atom may move to any valid cell in its Moore neighborhood. For a nonbonded atom, a cell is only invalid for movement if it is not empty. However, additional constraints apply to bonded atoms. While moving, a bond is not allowed to stretch (move outside a Moore neighborhood of radius 2) or cross (intersect other bonds). The new cell of an atom is randomly selected from a list of valid positions (including the current position of the atom). These two constraints allow for the formation of appropriate membranes. A closed chain of linked atoms is *selectively permeable*: such a membrane allows nonbonded atoms to pass through at a reduced rate, while blocking bonded atoms. In practice, this means that food can easily enter the membrane, and enzymes may be secreted, while diatomic enzymes or larger structures are contained by the membrane. It is worth noting that, though Hutton claimed the two conditions of bond crossing and stretching were sufficient, we have found that a slight modification is necessary: the movement path of a bonded atom must not be allowed to cross other bonds. Without this constraint, the membrane may move around a free atom, causing a diatomic molecule to pass through a membrane.

Formally, this discrete-space implementation could be considered a nondeterministic cellular automaton: cells proceed through a series of discrete states in a process determined by neighboring cells. However, this definition is not fully satisfying. Consider a situation in which an atom may move to any of four adjacent cells. A formal definition in the language of a cellular-automata would require each of the four adjacent cells to communicate, though some hidden mechanism, for a neighboring cell to adopt the state which represents the moving atom, and the central cell to adopt an empty state. On a fundamental level, a cellular automata definition treats cells, not atoms, as the unit of interest—obfuscating the defining principle of conservation of matter and excessively complicating movement. More complications arise in a system with multiple atoms, as our implementation is asynchronous: atoms are moved one at a time, a situation fundamentally different from traditional cellular automation. Though such a system could be simulated with a complex system of cellular automata rules, such a system would be unwieldy. For these reasons, we will not consider this system a cellular automaton, though we will soon see that features and terminology of cellular automata may be applied to our system at a higher level.

4.1 A basic organism

Now we will turn our attention to the design of the initial organisms within this system. The majority of this design comes from Hutton, with the exception of enzyme translation mechanisms, which were rewritten for our new enzyme system. However, many reactions have been simplified away, primarily with the use of three-way reactions. The original types and states used have been preserved wherever possible, for comparative purposes (this leads to gaps in the numbering of the states used).

Basic organisms have three components: the *membrane*, the *DNA* and the *cytoplasm*. The membrane, which encloses the organism, is a closed chain of A36 atoms. The membrane is attached to the DNA strand at two A37 *poles*.

The DNA strand contains the reaction data of the organism. It is a linked chain of atoms between the two membrane poles which divides the organism into two sections. The beginning of the chain is an atom of type E, the end an atom of type F. In between is an assortment of atoms of varying types, each of which store reaction data. The types of these atoms correspond to a particular mode of expression. Type-A atoms embed enzymes in the membrane. Type-B atoms create diatomic enzymes which are contained in the cytoplasm. Type-C atoms create nonbonded enzymes which may pass through the membrane.

The remainder of the space of the organism is the cytoplasm. The cytoplasm contains free-floating molecules, primarily enzymes and food.

4.2 The lifecycle of the organism

In order to understand the DNA of our organisms, it is helpful to abstract different reactions into *gliders*. A glider, a term borrowed from cellular automata, is a signal which moves across a medium by repeated reactions, possibly effecting some change in the environment. The standard lifecycle of the organism consists of five distinct gliders, all of which propagate along the DNA strand, and assorted unassociated reactions. We will refer to these gliders as *G1, G2, ... G5*. We can also consider a different grouping of reactions: A *multipart reaction*. These reactions could be represented as a single reaction between n reactants. As our system only allows for reactions between two or three atoms, these multipart reactions are instead represented as a set of individual reactions. Using these two reaction groupings, we can describe cellular chemistry on a high level.

The first glider, *G1*, mediates DNA replication. Starting at the E-pole, G1 proceeds down the strand. It forms a parallel strand of DNA attached to the original, absorbing free food atoms and copying over the reaction data of the DNA.

R1	E10 - A37 + E0 + to 2 - 37 - 3 -	Incorporates a second initial E atom and begins G1
R2	X3 - X2 - Y1 + to 7 - 2 - 5 -	Propagates G1
R3	X5 + X0 to 6 - 6 (cpy)	Adds atom for parallel DNA strand
R4	X7 - Y6 - Y6 + to 2 + 2 - 3 -	Connects newly incorporated parallel DNA atom
R5	F3 - F2 - A37 + to 8 + 8 - 11 -	Bounces G1 into G2
R6	X2 - Y8 to 9 - 1	Propagates G2
R7	X9 - Y9 to 8 + 8	Splits the DNA strands
R8	E8 - A37 to 1 - 37	Absorbs and dissipates G2
R9	F1 - A11 - A36 + to 13 - 11 - 37 -	Prepares the membrane to be pulled
R10	A11 - X13 - Y1 + to 11 + 17 - 13 -	G3
R11	E13 - A37 - E13 + to 18 - 20 - 18 +	G3 bounces into splitting reaction
R12	A20 - E18 - A11 + to 21 - 18 - 22 -	Joins last atoms of membrane after G3
R13	E18 - A22 to 31 + 23	Splits DNA from one pole
R14	E18 - A21 to 31 + 24	Splits DNA from one pole
R15	A36 - A24 - A23 + to 36 + 25 - 23 -	First part of splitting reaction
R16	A37 - A23 - A25 + to 37 + 30 + 30 -	Second part of splitting reaction
R17	A30 - E31 to 37 - 32	Reforms the pole after splitting
R18	X32 - Y17 to 41 - 32	Propagate G4
R19	B41 + B0 to 42 - 43 (cpy)	Binds a free atom for a B-enzyme
R20	B43 + B43 to 44 - 44	Joins two B-enzymes to make a diatomic enzyme
R21	B43 + B0 to 44 - 44	Joins B-enzyme with food to make a diatomic enzym
R22	B42 - B44 to 45 + 50	Releases a B-enzyme from DNA strand
R23	A41 + A36 to 45 + 36 (cpy)	A-enzyme transcription
R24	C41 + C0 to 45 + 50 (cpy)	C-enzyme transcription
R25	F32 - A37 to 46 - 37	Bounce G4 into G5
R26	X46 - Y45 to 1 - 46	Propagate G5
R27	X46 - E41 - A37 + to 1 - 10 - 37 +	Bounce G5 into G1
R28	A37 - A37 + A0 + to 37 + 37 - 36 -	Adds atom to membrane
R29	A37 - A36 + A0 + to 37 + 36 - 36 -	Adds atom to membrane
R30	A36 - A37 - F11 + to 37 - 36 + 11 -	Pole moving reaction
R31	A36 - A37 - F8 + to 37 - 36 + 8 -	Pole moving reaction
R32	A36 - A37 - F1 + to 37 - 36 + 1 -	Pole moving reaction

Table 4.1: All the reactions in a basic organism

1	The base state of DNA molecules
2	The leading edge of G1 and G2
3	The leading edge of the second DNA strand during G1
5	The edge of the DNA strand which accepts new atoms during G1
6	The edge of both strands of DNA after a new atom has been absorbed
7	The edge of the replicating DNA strand ahead of the absorption of new atoms
8	DNA about to split during G2
9	DNA after splitting during G2
10	E10 signals the start of G1 and the beginning of a new cycle
11	Lower pole during membrane pulling
13	DNA attached to the membrane pole during membrane pulling
17	DNA state after membrane pulling
18	State of E atoms at the start of the splitting reaction
20	State of E pole at the start of splitting reaction
21	State of E pole during splitting reaction (after 20)
22	State of F pole during splitting reaction
23	State of F pole after one strand splits off
24	State of E pole after one strand splits off
25	Intermediate state of E pole during multipart splitting reaction
30	Membrane which will form the edges of the new cell membranes
31	State of E atoms during splitting reaction
32	Leading edge of G4
36	Basic membrane state
37	Basic state of membrane poles
41	DNA atom which will absorb a new atom to create an enzyme
42	DNA atom which is currently forming an enzyme
43	Newly created enzyme which is waiting for a pair
44	Enzyme which has found a diatomic pair but has not been released
45	DNA atom which has completed enzyme transcription
50	State of an enzyme

Table 4.2: A rough interpretation of the states in the cycle of a standard organism

G1 bounces off the F-pole, creating *G2* and *G3*. G2 proceeds backwards from the F-pole, splitting the double DNA strands formed by G1. After G2, there will be two unconnected identical strands of DNA, each joined to both poles.

G3 follows behind G2. During this reaction, the pole of the membrane is passed through the separated strands, partitioning the cell. When G3 reaches the opposing pole, a multipart reaction splits the cell and establishes the new poles for the two daughter cells.

After the cell is split, *G4* proceeds from the E-pole to the F-pole. G4 places every atom in the DNA strand in state 41. These state-41 atoms react with food atoms to form enzymes, according to the type of the DNA atom. After enzyme formation, the DNA is placed into state 45. When G4 reaches the F-pole, it bounces back into *G5*. *G5* proceeds back to the E-pole, only passing though the state 45 atoms which have finished enzyme formation. When all enzymes are formed, and G5 reached the E-pole, G1 is reformed and the lifecycle begins again.

The glider-based approach provides an insight into the structure of our organisms which will be more fully explored in later chapters.

4.3 Membrane Behavior

There are a number of interesting and complex behaviors displayed by the membrane.

Constant growth is essential to the membrane. Hutton used a reversible three-way reactions to add and remove free atoms from the membrane. This produced a membrane roughly the size of the organism [8]. However, we have made other changes which make this solution problematic. The increase in reaction range causes an overexcited shrinking effect, and enzymes embedded in the membrane may be removed. Therefore, while atoms are still added to the membrane by a normal enzyme, a hardcoded reaction is needed to shrink the membrane. This mechanism will split off an atom in a chain of linked A36 atoms if the two bonded atoms are adjacent. This mechanism will not effect enzymes or atoms with more than two bonds.

Consideration must also be given to the location of the poles of the membrane. Hutton depicts the two poles residing on opposite ends of the membrane, resulting in a straight DNA strand [8]. Directly after division, however, the two poles are adjacent, forcing the DNA strand to curl into a horseshoe. Hutton shows natural membrane growth and decay separating the two poles, but we have found that this does not happen reliably. If the two poles are close when G3 occurs, the organism reaches a deadlock. As the membrane is pulled through the split DNA strands, the outer strand is pulled tightly against the inner strand, until neither strand can move, causing the deadlock. To prevent this deadlock, we add a series of pole-moving reactions. These three-way reactions will change the position of the poles randomly,

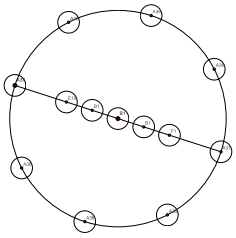

Figure 4.1: The initial stage of Hutton's organism

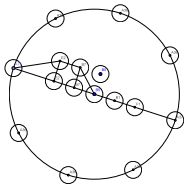

Figure 4.2: The strand of DNA absorbing free food atoms during replication

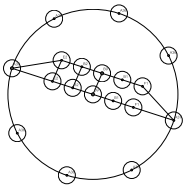

Figure 4.3: The strand of DNA is beginning to split into two strands

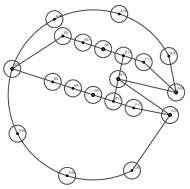

Figure 4.4: The membrane is being pulled through the separated DNA strands

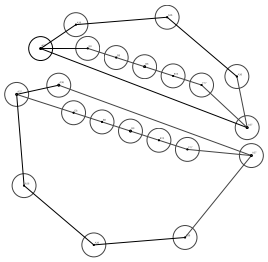

Figure 4.5: The organism has split and is ready to divide again. Note the adjacency of the two poles.

reducing the change that a deadlock will occur, and quickly resolving most deadlocks. However, it is not possible to perform these reactions when the DNA is split, as four atoms would be involved. This limitation allows a rare possibility for deadlock.

Even with these reactions, the poles tend to remain close to each other. This effect leads to an uneven division of the cytoplasm. In most casts, one daughter cell will inherit almost all of the enzymes. Over time, this creates a sharp divide between massive and small organisms.

4.3.1 Summary

We have described the physical systems which underline our organism. In general, the fact that such specific physical effects arise from our basic rules shows the power of our rules themselves. However, we also must acknowledge the inherent problems which arise out of this system, some of which will be addressed in the next chapter.

Chapter 5

Our Modifications

Having described the basics of our system, we will now explore the rationale behind many elements of our system, especially when we differ from Hutton's system.

5.1 Reaction Data

Our primary divergence from Hutton's system is the use of reaction data. We have detailed our version of enzyme action above, and we will now explore Hutton's version in greater detail.

5.1.1 Hutton's enzymes

In Hutton's system, the genome of an organism is encoded as a sequence of types of atoms in the DNA. Genes (which correspond to one enzyme) are delineated by type-f atoms. During enzyme translation, each gene is resolved to an integer value, which is the state of the enzyme.

Informally, the gene is read as a base-4 integer. Atoms of type $\in \{A,\ B,\ C,\ D\}$ correspond to $\{0,\ 1,\ 2,\ 3\}$ respectively. This integer is offset by 38, so that the state of the enzyme is distinct from other components of the organism. (In Hutton's system, all other components have states ≤ 37, so every atom with state ≥ 38 is considered an enzyme. This restriction is not present in our system.)

The state of an enzyme to catalyze a given reaction is determined by the following form. The bond between the reactants is represented as an integer (0 or 1). Similarly, the types are converted to integers. The state which will represent a given reaction is found by combining all of the integer values which make up the reaction by adding each successive value and multiplying by its maximum value.

5.1.2 Advantages

The primary appeal of Hutton's enzyme system is its similarity in structure to biological life, both stylistically and conceptually. Hutton's enzymes and the process of enzyme translation obey the *central dogma* of biology: information flows from DNA to enzymes. In other words, Hutton's system creates a meaningful distinction between genotype and phenotype. Hutton's DNA strand is only able to store and replicate information and to pass that information on to enzymes. Crucially, the format of the information of the DNA strand differs greatly from that of the enzymes. In contrast, in our system, DNA is, by itself, equally catalytic as enzymes. However, it is unclear what, if any, role this behavior plays in evolutionary growth. In particular, Taylor proposes that a distinction between genotype and phenotype is essential for evolutionary growth [13]. Much of his argument relies on a flexible interpretation from the genotype to the phenotype: if the interpretation itself could evolve, the interpretation mechanism could become more favorable to evolutionary growth (as is seen in biological mechanisms such as exon shuffling).

Hutton's enzymes, however, require a hardcoded mechanism. The base-4 mechanism central to enzyme translation cannot be carried out by normal enzymes, and so it cannot be modified by mutations. In contrast, our system is completely self contained, as the mechanisms for copying data from DNA into enzymes is simply contained within the copying flag of reaction data. The requirement of a hardcoded mechanism negates any advantage gained by a flexible interpretation.

The primary advantage of our system is its compactness. Hutton's system requires a string of DNA around 15 atoms long to encode a single enzyme. The result is that encoding all 41 enzymes required for cellular reproduction is not computationally feasible. Instead, Hutton hardcoded all of the reactions into a universal reaction table. These reactions operate independently of enzymes. Though modified enzymes which create divergent behavior may arise, any modified enzymes must be mutated from scratch, not derived from existing genes. This seems unlikely. Even if such enzymes were to emerge organically, they could still not override the hardcoded enzymes which direct the growth of the cell. As enzyme-mediated reactions require that the necessary enzymes are present, they would have a much lower frequency of activation than a competing hardcoded reaction. The result is that most of the organism's lifecycle cannot be effectively overridden from the hardcoded behavior. The increased compactness of reaction data allows us to encode all the previously hardcoded reactions in the genome of our organism with a relatively short DNA string, removing these problems.

Our reaction data also allows us to create more efficient mutation mechanisms. Mutation in Hutton's system is done by randomly adding or removing atoms from the DNA. The mechanism of interpretation of genes into enzymes means that a mu-

tated gene will likely be completely different from its original. There is no room for incremental mutations, meaning all mutation is a random, not guided, search. In our system, we can have much finer control of the mutation function which is written in code. This allows more sensible or potent mutations, which we will explore in later sections.

There are additional possibilities opened up by our use of reaction data. The ability to embed many enzymes in one atom permits greater diversity of behavior. A single atom, released from the cell, may now act as a powerful and complex agent, allowing viruses or horizontal gene transfer to emerge.

5.2 3-way mutations

Hutton used a handful of 3-way mutations in his initial genome, though they were limited to membrane growth and enzyme translation. We have expanded their use, and used them wherever possible in our modified genome. A complex series of 2-way reactions can often be replaced by a single 3-way reaction. 15 out of 32 enzymes in our genome are three-way reactions. For instance, consider the reactions which split the two cells during cell division:

R27	a24 + a37 to a26 - a27	Attaches the two poles of the cell A
R28	a27 - a23 to a37 + a28	Breaks bond between the second poles of cells A and B
R29	a26 - a36 to a29 - a30	Communication
R30	a30 + a28 to a25 - a33	Attaches the first pole of cell B to its own membrane
R31	a31 - a25 to a32 + a36	Breaks bond between pole of cell A and membrane of cell B
R32	a32 - a30 to a34 - a36	Communication
R33	a34 - a33 to a37 + a37	Breaks last bond between two cells

Table 5.1: The 2-way reactions used by Hutton to perform the final split of the daughter cells

Compare this to the same process rewritten using 3-way reactions

R15	A36 - A24 - A23 + to 36 + 25 - 23 -
R16	A37 - A23 - A25 + to 37 + 30 + 30 -

Table 5.2: The same reactions rewritten with 3-way reactions

3-way reactions provide a conciseness in application. It is worth examining where this brevity arises from.

Note the reactions marked as *communication* in Table 5.1. These are reactions needed to synchronize and coordinate between atoms. For instance, a communication

reaction might pass a signal that one reaction has been completed and another one may now be started. We can loosely categorize all reactions that do not break or form bonds, or copy reaction data, as *communication reactions*. These reactions can almost entirely be seen as inefficiencies. Most notably, these reactions, which contain many delicate parts sensitive to disruption, are difficult to mutate effectively. Simpler, non-communication reactions have a definite effect and purpose, and are generally more robust. Communication reactions have more context-dependent effects, a problem which will be discussed in later sections.

Hutton's original genome contains 18 reactions (of 41 total reactions) which do not break or form bonds, and could be considered communication. Our system contains 7 reactions (of 32 total reactions) which could be considered communication. This lack of communication reactions leads to a more efficient, robust and mutable genome.

It should be noted that we are not trying to limit communication itself, or the sharing of information across the organism. Many imaginable communication reactions could act behaviorally, transmitt information or make decisions. However, the communication reactions we see in our present organisms have none of these properties. Instead, communication reactions primarily represent inefficiencies in complex reactions.

Looking at our modified genome, we can find a few examples of reactions which may be eliminated if 4-way reactions were allowed. For example, the splitting of the two DNA strands. Others would become simpler with arbitrarily many reactants, such as the splitting off of daughter cells discussed earlier. One might consider expanding 3-way reactions into *n-way* reactions. However, this creates several severe disadvantages. The first is the significantly increased computation time: 3-way reactions already require a significant increase in computation time, and larger reactions would increase this computation exponentially. Multicomponent reactions are also triggered less frequently, as all components must make contact at once. Here, we see a tradeoff between computational difficulty and evolutionary simplicity. We have opted to allow 3-way reactions but have avoided more complexity.

5.3 Basic Physical changes

We have made a number of changes to the physics of the environment in order to speed up cellular reproduction. Our hope, of course, is that these performance improvements do not come at the cost of stifling evolution.

Perhaps the most drastic change has been to increase the radius of reactions. Previously, two atoms must be in a Moore neighborhood of radius 1 to react. We have increased this to a Moore neighborhood of radius 2. This change clearly increases the rate of reaction, as components find each other more frequently. It has special

effects on two key areas. The first is reactions between two bonded atoms. Remember that bonded atoms may move within a Moore neighborhood of radius 2 of each other. Previously, two bonded atoms for which a reaction exists may not react for some time, if they never moved closer. Now, bonded atoms interact almost instantly in most circumstances. This has a drastic effect on the speed of self-propagating reactions, such as the splitting of the DNA strand. Our change also reduces a critical bottleneck in the lifecycle of the cell: when the membrane is pulled through the DNA, it must reach the pole on the opposite side. Due to the orientation of the membrane, waiting for both poles to connect was a very time-consuming event which often consumed the majority of time in the life cycle of the cell. While increasing the reaction range significantly shortens the iterations per cell division, it carries a tradeoff in computational time, as more cells must be searched for reactants.

We have also made some changes to the movement rate. In our first simulations, atoms would have a certain chance to attempt movement. Attempted movement would then be checked against the physical constraints discussed earlier (bond crossing and bond stretching) as well as the presence of another atom in the target location. The result was that highly-bonded or tightly-packed atoms moved less frequently. Bonded atoms would display a natural tendency to stretch their bonds, as two atoms would be in a significantly more stable position at a distance of 2 instead of a distance of 1. Instead, we now compile a list of all valid movement positions of a given atom, including its current position (no movement), and randomly select one. The result is more natural movement, and fewer deadlocks, but this change adds another cost to computational time.

A third change is a slight fix to Hutton's bond-crossing constraint. We prevented the diagonal movement of a bonded atom from crossing any bonds. Without this constraint, a diatomic molecule (such as an enzyme) may pass through the membrane as the membrane moves around it. Due to the necessity of this change, it is almost certain that this constraint is present, but undocumented, in Hutton's simulation.

5.4 Membrane Physics

A wide variety of physics effects can be seen by looking at membranes.

First, we must look at the growth of the membrane. Hutton depicted a reversible 3-way reaction which added atoms to the membrane. This would cause the membrane to be slightly bigger than the contents of the cell. We had some difficulty, likely due to the other changes we have made, reproducing this behavior in a suitable manner. Our solution is multifaceted. First, membranes (strings of A36 atoms) lose atoms under certain conditions: The atom in question is not an enzyme and is bonded to two A36 atoms, that are adjacent to each other. This behavior is hardcoded, but can

be easily avoided by organisms by placing their membranes in a different state.

Hutton depicts the pole positions in the initial cell directly opposite each other. However, after one round of cell division, the poles are adjacent to each other. This change is not merely topological. The poles are bonded to each other. Visualizations by Hutton (in continuous physics) show these poles moving effortlessly apart, though we have found that this does not reliably occur. If the poles are adjacent during membrane pulling, serious problems occur. The adjacent poles cause the DNA to take on a horseshoe shape. As the membrane is passed between this horseshoe, the outer strand is pulled quickly. This continues until the outer DNA strand and the membrane tightly encircle the inner DNA strand. As a result, neither the outer or inner circles can move, or can move only trivially, causing a complete and fatal deadlock. The addition of free food atoms is softcoded, and only occurs around the poles. This allows some Type-A food atoms to pass into the cell, and helps to combat these deadlock problems.

A variety of methods are used to reduce the frequency of this deadly event. First, as mentioned before, new membrane atoms are added only near the poles, causing more pressure for the poles to move apart. Additionally, we add a series of supplementary 3-way reactions, R31-R33, that (randomly) move the poles of the membrane. While these changes cannot eliminate this deadlock, they can significantly reduce its occurrence to negligible levels.

5.5 Enzyme Physics

Hutton's enzymes must move adjacent to the affected reactants in order to catalyze a reaction. This quickly becomes impractical when all reactions are catalyzed by enzymes. Instead, we allow enzymes to catalyze reactions between all atoms in a range of ten cells. However, this change carries its own problems.

Hutton has previously demonstrated the parasitic effects caused by enzymes which are not contained by a membrane (this was the rationale for the original implementation of membranes). Positive effects of enzymes will be passed on to neighboring cells. Negative effects, usually fatal, will also leak through the membrane.

To prevent these effects, we implemented restrictions on the effect of membranes. Enzymes can reach reactants if either the line between the reactants and the enzyme does not cross a membrane (any bond between two type-A atoms) or if the two atoms are directly or indirectly connected. Atoms in a membrane (defined as type-A atoms with $30 \leq state \leq 40$) must be connected to the target enzymes.

This creates an incentive to embed enzymes in membranes. We define three types of enzymes, differentiated by their targets. Each enzyme is marked by the type of atom in the DNA which contains that enzyme. *Standard enzymes*, which target the

interior of the cell, are marked by type-B atoms. These enzymes are expressed as diatomic molecules contained by the membrane. *Membrane enzymes*, which target the membrane itself, are marked by type-A atoms and copy their reaction data to an atom in the enzyme. The final type of enzymes, of which none exist in the standard cell, are *secreted enzymes*. These enzymes are non-bonded, and can pass through membranes freely.

5.6 Degradation

Naturally, a population of rapidly reproducing organisms will eventually exhaust all the available space and food. As a constantly expanding map is not computationally feasible, new avenues for growth must be opened through floods. In Hutton's simulation, floods affected a randomly selected quadrant, breaking any bonds between the affected atoms and setting their state to 0. However, unintended effects may arise from the positioning of floods. If floods only affect quadrants, and organisms do not have significant net movement, organisms inside the interior of each quadrant have little chance of passing on their genome long-term. The only viable genetic lines are those which occur on the edges of quadrants. In effect, most computational power is wasted simulating doomed genetic lines in the interior of the quadrant. To counter any possible effects of quadrant-based floods, our floods target a randomly selected area. Our floods affect a square area of radius 20, centered on a randomly selected cell.

Hutton's floods affected not only any atoms in the flood range but also any atoms recursively connected to affected atoms. This behavior means that entire cells vanish in a flood. Our floods do not extend beyond the affected area, meaning that the area of the flood often splits organisms, resulting in partial organisms. These partial organisms are cleared by *caustic agents*.

Caustic agents were originally introduced by Hutton in an auxiliary paper [9]. Caustic agents will degrade all atoms which they contact. Their effects are blocked by membranes (type-A atoms with at least 2 bonds), and they exist as diatomic molecules so that they cannot cross membranes.

Besides the degradation of organisms, caustic agents serve several crucial evolutionary purposes. First, they reward the creation and maintenance of membranes, which would otherwise offer little benefit for their cost. They reduce, though do not eliminate, the effectiveness of secreted enzymes, helping to regulate secreted enzymes. Ideally, each secreted enzyme would interact with a small number of cells, on average, while not persisting forever. Caustic agents provide an appropriate limitation.

5.6.1 Summary

We have detailed a great deal of physical changes to Hutton's system. We have provided justification for our modifications. However, the real benefit of our individual changes remains difficult to experimentally test, due to the complex nature of success in our simulation.

Chapter 6

Mutation

The implementation of reaction data allows for considerable control over mutation. Mutation in Hutton's system is unsatisfactory in several ways. As discussed earlier, random mutations in Hutton's system produce essentially random enzymes. This eliminates any chance for incremental progress. There is another hidden cost to Hutton's mutation: it only affects strings of atoms in state 1. It would be trivial to build an organism which does not mutate by replacing state 1 with an otherwise equivalent state. In fact, some alternative organisms we might wish to simulate, such as smaller viruses, would have no mutation mechanism at all.

Our system solves the second problem by tying mutations to the copying of reaction data, a fundamental process we can expect every self reproducing organism to use. The mechanisms of mutation allows us to address the first problem. As data is stored in code, and not in the environment, the mutation code can have almost complete control over the mutations formed. As we will describe, this system allows for considerable improvements to be made in the mutations as compared to those of Hutton.

6.1 Problems

We will first describe the problems a system of mutations must address.

The first problem is the enormous number of possible enzymes. For $state \leq 100$, our system allows approximately 51 trillion 2-way reactions and $6.6 \cdot 10^{16}$ 3-way reactions (although many of these reactions are functionally identical). Considering the computational time required to simulate each cell over even one life cycle, exploring all of these possible reactions to find the relevant reaction is not feasible. In addition, the vast majority of these enzymes have no expressed effect. These enzymes describe reactions between reactants which will never collide in the proper type and state. Of

those enzymes which do have an effect, the majority are immediately fatal to the cell. If the products produced are not the reactants for the following reaction, and the atoms in question are necessary to the cell (as are most atoms), the cell will not be able to reproduce successfully and will eventually be degraded.

The second problem is the lack of *silent* or *incremental mutations*. Very few mutations produce any difference in fitness. As mentioned above, almost all enzymes are either fatal or have no effect. In simple terms, either an organism is able to successfully reproduce or its enzymes malfunction and the organism will fail. Occasionally, a mutation might result in a significant increase in fitness, in which case it would become dominant. Crucially, there are no quantitative mutations. The fitness function of any quantitative variable would create exactly the kind of mostly smooth curve we are looking for, but our present boolean genes fail to yield any smooth curve.

Though most enzymes have no effect, there can still be a lack of silent mutations. Almost all of an organism's enzymes code for a necessary reaction, so a mutation from a required enzyme to a neutral one is not silent but fatal. In order to encourage silent mutations, we can encode redundant copies of each enzyme in the genome. While one copy mutates, the second copy can direct the reproduction of the cell as normal. This approach is similar to the double chromosomes and recessive traits of biological organisms.

A third and much more concerning problem is the potential lack of beneficial mutations of our initial organisms. As described previously, there are few quantitative shifts which could occur from our initial organisms, and it is hard to imagine more than a handful of quantitative changes. If all except for a select number of beneficial mutations are inaccessible to mutation from initial cells, evolution will stagnate. This is by far the most serious potential problem, as well as the most difficult to test. Fixing this problem would involve changing either the system or the initial organisms themselves. Very simple organisms might produce evolutionary activity we do not see with more engineered species. Similarly, organisms which display more complex behaviors, such as the secretion of hostile enzymes, might encourage the development of defensive enzymes in other organisms.

6.2 Languages

The problems described above arise in part from the language in which enzymes are expressed, and, therefore, in which enzymes are mutated. We consider the *language* to be both the notation we use and the mutations arising from that notation. Our current language, which describes an organism in terms of reactions between atoms, struggles to holistically describe the larger processes of an organism.

Consider the search space problem described above. As previously shown, there

are an extremely large number of enzymes which can be expressed in our language. This number grows exponentially when small groups of reactions are rightfully considered as a functional unit. However, if we look at a (loosely defined) notion of *behaviors*, qualitative structures or actions coded for by some number of enzymes, we would expect to find that the number of possible behaviors is many orders of magnitude below the number of possible enzymes. The result is that the vast majority of sets of enzymes code for identical behaviors or, more frequently, for no behavior at all.

A massive search space would be manageable, possibly even desirable, with the aid of a directed evolutionary search. However, Hutton's system does not possess such a directed search. As shown above, Hutton's base-4 system is not a sufficiently powerful directed search. We could implement a basic mutation scheme, where a single type, state or boolean flag variable in a reaction selected for mutation will be changed. This might appear to be a directed search, as the mutated enzyme would look similar to the original. However, almost all changes (with the exception of a change to some boolean flags) would create phenotypically dissimilar enzymes. That is, there is a massive difference between the original enzyme which catalyzes the absorption of food into a DNA strand:

$$X\mathbf{5} + X0 \text{ to } 6 - 6(cpy) \tag{6.1}$$

and its mutated form:

$$X\mathbf{6} + X0 \text{ to } 6 - 6(cpy) \tag{6.2}$$

The effect of a slight change in the mutated enzyme is a completely different effect (which will almost certainly be fatal). In fact, it is difficult to find a single mutated form of this enzyme with any qualitatively similar phenotypic effect. Conversely, this enzyme, and all similar enzymes, cannot be efficiently reached by mutation except from mutation from phenotypically different enzymes (that is, by pure chance). This leads us to our first desire for a language: that linguistically similar genes tend to be phenotypically similar.

Perhaps the most harmful feature of the enzyme language in this regard is the reliance upon states. This state-based system is fundamentally fragile: each component depends on each other component. With few exceptions, changing one enzyme in nontrivial ways requires that all associated enzymes be changed. The net effect of states is that the phenotypic effect of every enzyme cannot be isolated from the rest of the genome. While this behavior may be desired in some circumstances, it makes mutation extremely fragile. This gives us a second goal for an enzyme language: Each enzyme should have a relatively isolated phenotypic effect. These two statements are

far from the only properties we would desire from any language, but they form a useful basis for looking at the problems with our present system.

6.3 Other Languages

We have explored two alternative ways of categorizing behavior in our artificial chemistry which could both provide insights into the organization of our organisms and form the basis for a more robust formal language.

6.3.1 Gliders

The first relevant perspective is the use of *Gliders*, as described in Chapter 2. We described gliders as a self-propagating reactions along a medium, usually the DNA strand. As described in Chapter 3, the standard genome can be divided into five distinct gliders. In addition to these gliders, we also recognize a series of non-glider reactions, the most complex of which is the final fusion of the membrane. These multipart reactions could be simplified into a single reaction between n reactants, but must be expressed in our current system through multiple smaller reactions.

We can discuss a glider-oriented language in informal terms. We will use G1, the glider which duplicates DNA, as an example.

The glider is initiated by a *trigger*. This trigger can be defined simultaneously as either a specific arrangement of atoms (in the case of G1, an E10 atom joined to the pole) or the result of another glider (in G1, the completion of G5 leads into G1). For the reasons described above, the second approach, which largely avoids a reliance on states, is more easily mutated effectively. However, one can imagine behaviors which could only be expressed with the first method. In particular, gliders which fall outside of the normal reproductive cycle might require a state-based trigger.

The essential core of a glider is its *propagation mechanism*. This is defined as the interlinked tasks of propagating the glider through the medium and effecting some change in the medium. Due to the wide variety of complex effects, it is difficult to create a formal language to perfectly capture the effect of a glider on its environment. Informally, we can consider a *pre-glider medium*, the state and arrangement of the medium required for the glider to propagate, and a *post-glider medium*, the state and arrangement of the medium after the propagation of the glider.

Finally, we consider the end of the glider, which can generally be represented as part of the trigger of the next one plus some effect on the edge of the medium.

While gliders represent an effective way of examining and grouping the reactions of our genome, they have several limitations that prevent their usage as a complete

language. Even in our genome, they can only describe the standard reproductive life cycle. The membrane-controlling reactions cannot be classified well as gliders.

More seriously, we can easily imagine that alternative organisms will not easily fit this classification. Our genome fits this model so well because the reproduction of our organisms is centered around changes in a variable-length genetic strand. We can imagine other classes of organisms designed very differently, which would not be easily represented in a language based on gliders. Reliance on gliders may impede the fundamental evolutionary shifts which we want to encourage.

6.3.2 State transitions

Another method for examining our genome results from looking at atoms and states.

In this model, we consider only the transitions made by an atom between different states. For instance, in the reaction

$$X5 + X0 \text{ to } 6 - 6(cpy) \tag{6.3}$$

we use only two pieces of information:

$$5 \text{ to } 6 \tag{6.4}$$

$$0 \text{ to } 6 \tag{6.5}$$

From this information, we create a graph which represents this genome. Each vertex on this graph represents a state, and each directed edge represents a transition from one state to another, as shown in Figure 6.3.2.

This is a very simplified model. For instance, we do not distinguish between atoms of different types, nor do we recognize that some reactions will never be used as the second reactant will never be present. This graph does not attempt to capture many of the complexities of a genome: all topological information, for instance, is disregarded completely.

However, this graph conveys some useful information about a genome. In particular, insight is gained by comparing cycles in this graph. Only one vertex in our graph has no outward edges: state 50 (the state of synthesized enzymes). We can say that any vertex with no outward edges usually represents a created or synthesized product. A mutation which breaks an established cycle usually translates into a fatal mutation: the atoms are not able to reproduce and return to their original state. It is important to note that these measurements are merely heuristics: while they work for the majority of cases, it is easy to find exceptions to these rules.

Consider the example of atoms in state 37 (the poles of the membrane). We can see that there are two possible cycles originating from state 37: the pole may

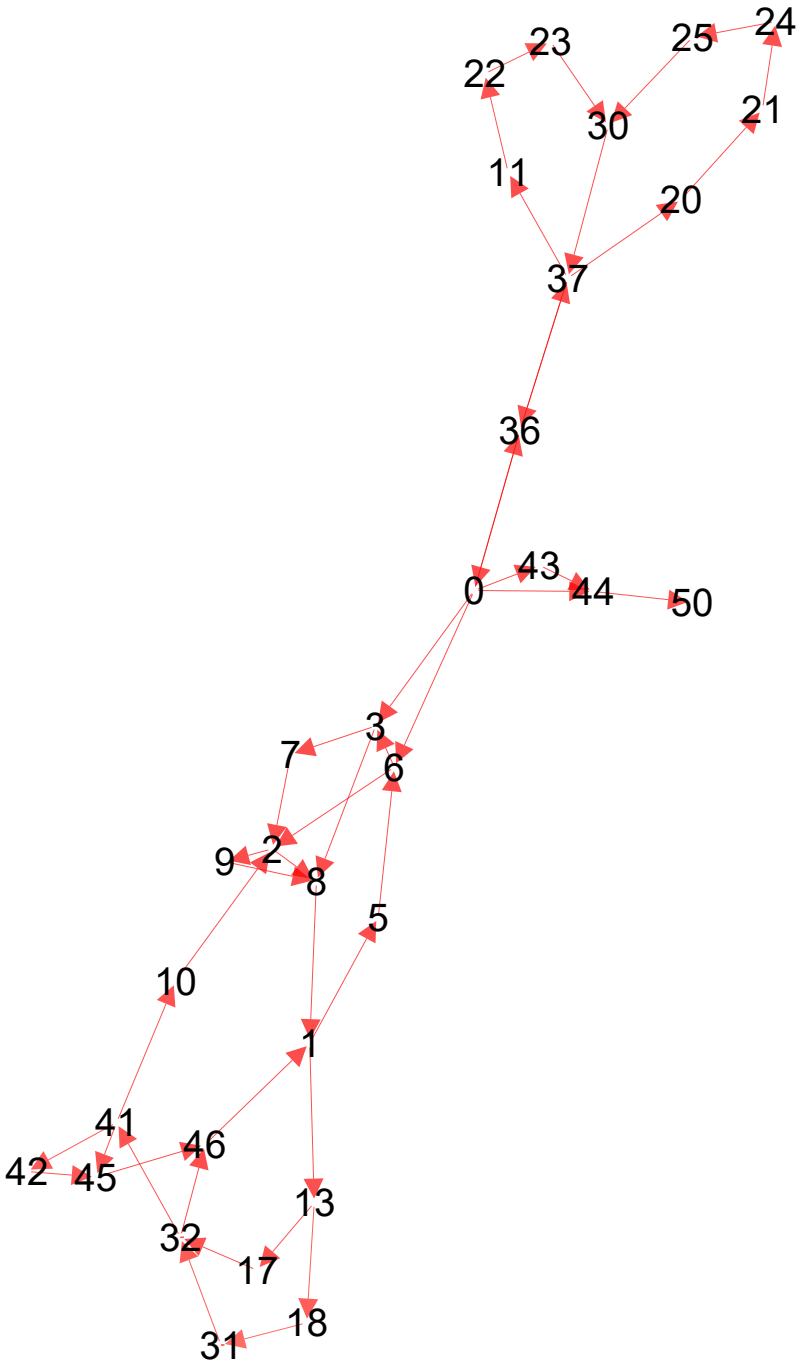

Figure 6.1: A visual representation of a state-based approach to understanding our genome

move into either state 20 or state 11. Reference to the original genome shows that state 11 represents the pole of the membrane pulled through the DNA, while state 20 represents the initial stage in the splitting of the organism.

This example also shows the limitations of this approach. A look at the graph may suggest that the pole may have a chance of moving into state 20 or state 11. Alternatively, it could suggest that the cycles are followed sequentially at different points in the reproductive cycle. Instead, the initial pole of the organism will always move into state 20, while the second pole will always move into state 11. In fact, much of the complexity and subtleties arises from these branching points: points where it is not initially clear from the graph which state will arise next.

As a corollary, isomorphic mutations which affect the graph but do not alter these branching points can be expected to alter the behavior of the organism while leaving its core structure intact—exactly the behavior we desire.

6.3.3 Our mutations

Using the insights gained from the preceding discussions, especially the graph-based approach, we have created a more advanced mutation system.

Whenever an enzyme is mutated, one mutation mechanism (out of 5) is picked at random.

The first three mutation mechanisms represent *basic mutations*, either of a state, type, or boolean flag value of a single reaction. Mutated state values are distributed normally around the original value, so that a mutated state value tends to be closer to its original value.

The fourth mutation mechanism is *simple duplication*. A reaction is copied to an empty spot on an enzyme. This mechanism is intended to allow for more silent mutations, as discussed above.

The fifth mutation mechanism, a *merge mutation*, is more complex. Treating the enzyme as a whole, this mechanism randomly selects two states from all the states present across the enzyme. All references to the first state are changed to references of the second state. This mechanism is created to attempt to create mostly-silent mutations which may have smaller effects on behavior.

6.3.4 Summary

The effectiveness of mutations is inherently linked to the language in which we write our enzymes. We have considered an alternative language, however we have not been able to fully define an alternative. Using the insights from this process, however, we have devised alternative methods of mutation, especially the merge mutation. The

effectiveness of these new mutations will be partially demonstrated in the following chapter.

Chapter 7

Experiments and Results

We have performed a series of experiments to test pieces of our changes, with various degrees of success.

7.1 Basic Procedure

Experiments are defined using a properties file, which describes the parameters of the simulation.

First, the initial state is read from disk. The stored state is a human readable file which describes both the genome and the physical structure of the initial cell. The stored state also contains the initial placement of caustic agents. Free food atoms, however, are not present in the stored state and are instead populated randomly. We have placed 100 caustic agents (or 50 bonded pairs) in the map. Simulations, by default, run on a 150x150 square grid.

Population of food is performed after reading the state from the file. Twenty percent of the cells of the grid are randomly selected without replacement. If there is no atom already present at the chosen cell, a food atom is placed there. The result is that the actual food percentage slightly less than twenty percent, as some spaces will be occupied by the initial structures. Note that it is not possible to place caustic agents with this same method, as a caustic agent might be placed inside a cell membrane, something we wish to avoid for its immediately destructive effects.

The types of food generated is a weighted average based roughly on the components needed for an organism. The two primary components of the organism, types A and B, each have a weight of 8 (38%). The two other components, types E and F, each have a weight of 2 (9.5%). Type C, which is not currently used but is referenced in our genome as a component of a secreted enzyme, has a weight of 1 (4.8%).

All degenerative forces are delayed for 50,000 iterations, so that a robust population may develop. Two actions are limited by this: floods and mutation. Without this protection, chance floods or mutations will often destroy early life. Trials show that 50,000 iterations is a reasonable value beyond which growth plateaus. After 50,000 iterations, floods occur once every 10,000 iterations and affect a square area with a radius of 20 grid units (this is approximately 1.8% of the total grid area, neglecting conditions on the edge of the grid).

7.1.1 Data Collected

Our program collects three types of data by default. We have found this to be sufficient for analysis.

The first data type is a complete capture of the map state every 10,000 iterations. This state can be loaded and examined by hand if necessary.

Secondly, a list of new enzymes is stored every 1,000 iterations. New enzymes are defined as reactions which did not exist in the initial state. This list includes the frequency of occurrence of each enzyme.

Finally, a metric called biomass is stored every 1,000 iterations. *Biomass* is defined as the percentage of all atoms which are non-food. Biomass is used as a rough indicator of the population of organisms. This measure is by far the easiest to measure and graph.

7.2 Experiments

We ran a number of experiments to test the viability of our modeling environment.

7.2.1 Initial Failures

Many trials demonstrated certain features required for a practical simulation. Almost all of these features have been discussed above. Among these examples are the frequency of caustic agents, delays on degenerative forces, addition of supplemental membrane reactions into the genome, and the size and frequency of floods. One caveat, however, is worth mentioning. Our initial genome included a supplemental reaction which would be performed many times. This reaction,

$$A17 + A37 \text{ to } 17 + 37(cpy) \tag{7.1}$$

would provide an additional avenue for copying membrane-bound enzymes, in an attempt to reduce gridlock in specific situations. However, once mutations were enabled, this reaction would be performed incredibly rapidly, producing a large number

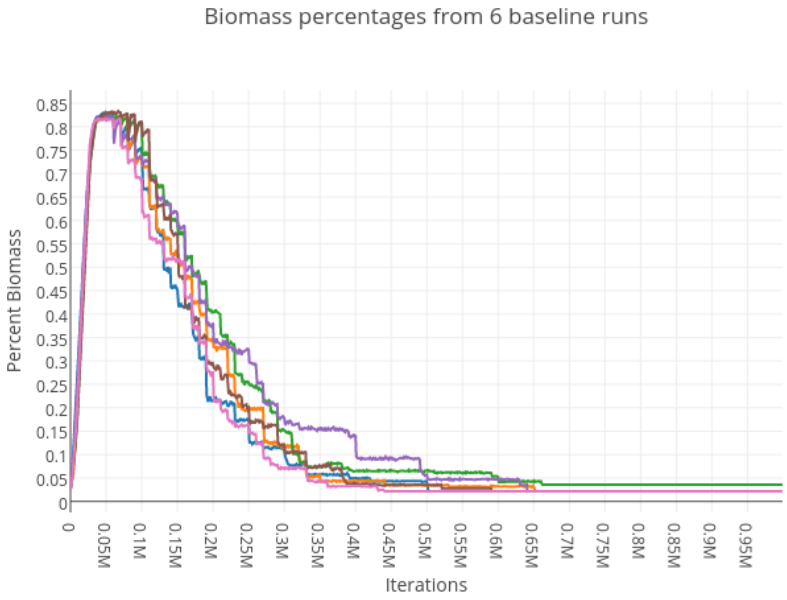

Figure 7.1: The results of six simulations with a mutation chance of .05. Each line represents a different simulation. The y-axis represents the percent biomass at different times during the simulation.

of mutations. This would proceed until one of the mutations prevented the reaction from being performed in some manner, often resulting in cell death. In general, we conclude that repeated copying of enzymes is quickly dangerous.

7.2.2 Basic Simulations

In experimenting with mutations, we ran a number of simulations, that exposed, to some degree, the results of the mutation mechanism. While we have learned some things, open questions remain.

In the first 50,000 iterations, the biomass increases exponentially until it reaches a plateau around 80%. At 50,000 iterations, as destructive forces are enabled, the biomass declines rapidly. The actual decline of biomass is primarily mediated by floods, the effects of which may be seen by the sharp drops every 10,000 iterations. Between floods, there is little biomass growth, especially at later iterations. Small amounts of population growth may be attributed to the absorption of free atoms by membranes.

To examine the cause of this decay, we consider the particular example of the first

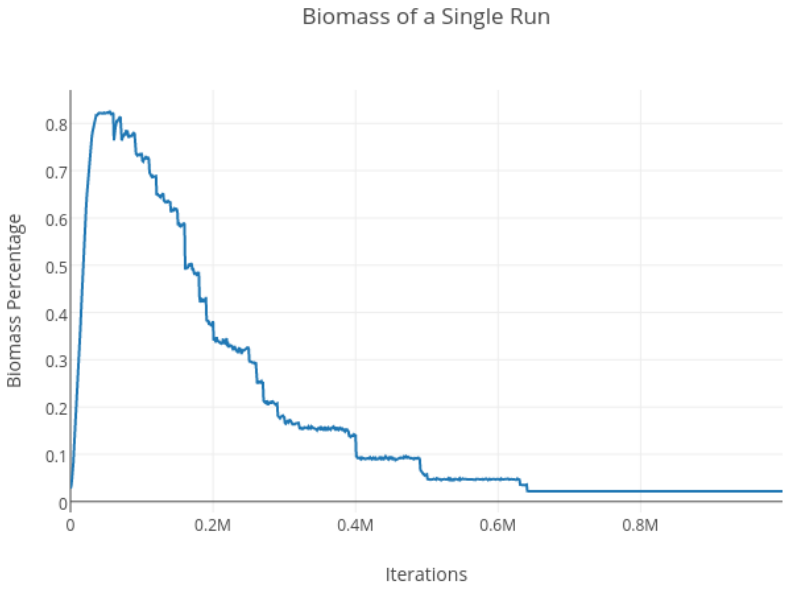

Figure 7.2: A representative line from 7.1, considering specifically the sixth run of the simulation

run of the simulation, paying particular attention to the developed enzymes. At the 100,000 iteration mark, At this point, 13 unique modified enzymes have appeared.

The first seven mutated reactions, all of which appear 47 times, are clearly the product of the merge mutation discussed earlier (in particular, a merge of state 37 into state 30). The next four mutations, each repeated 11 times (except for one, which is repeated 10 times), are another product of the merge mutation, this time from state 3 to state 0. The propagation of both of these sets of mutations strongly implies that the mutations are silent or mostly silent, as we would expect from our merge mutations.

The third set of mutations, repeated 3 times each, are the result of another merge mutation (state 37 to state 5). This mutation is likely partially fatal, as it would cause a membrane pole in state 5 will interact with free food atoms, likely catastrophically. As such, this cell may be able to reproduce once or twice before failure (note that as many as 3 instances of a reaction may result from a single cell, as produced enzymes are counted). The remainder of the enzymes, each of which only appear once, do not appear to have any catastrophic effects extending beyond the cell.

Considering that these mutations are likely contained within a few cells, only a small number of cells in total are likely to be negatively affected by mutations.

Simultaneously, however, our organisms display a complete lack of growth. In the 10,000 iterations since the last flood before these enzymes were observed, biomass increased only one percentage point. In the next inter-flood period, biomass actually declined slightly. This stands in sharp contrast to the exponential growth observed in the nascent period (in the 10,000 iterations before biomass plateaued at its maximum level, biomass increased 16 percentage points).

In short, we cannot find a reason why mutation contributed to the stagnation we observe. However, floods do not appear to be the cause of this stagnation: no significant growth is shown in inter-flood periods. We have no satisfactory explanation for the lack of growth at this time.

7.2.3 Lowering Mutation Rates

Our next simulations make two changes in an attempt to sustain a population. The first is the lowering of the mutation rate from 5% to 2%, in an effort to reduce the negative effects of mutations. For the same reasons, we also change the initial genome by adding a duplicate copy of each reaction, a change discussed in Chapter 5. The effects of these changes are summarized in Figure 7.3.

Two effects are noticeable from this graph. As before, biomass continues to decline after the 50,000 iterations mark. However, we now see a slower decline and periods of sharp growth, apparently interrupted by floods.

The distinguishing feature of this graph are periods of rapid growth do not share the same sharp slope as the pre-50,000 growth, indicating that the growth is either localized or relatively slow. Even more interesting are the sudden drops. These drops align with the 10,000 iteration flood periods, indicating that they were caused by floods. However, drops of up to 18 percentage points are observed, while floods only affect 1% of the total area of the simulation.

The cause of these growths can be seen by examining the actual state. As seen in Figure 7.4, the growth is caused by a single massive organism which is growing uncontrollably. The DNA of the cell is stuck and not reproducing (in the upper left corner of the cell). The growth is caused by some failure of the mechanism which regulates enzyme transcription. Production of more enzymes drives pressure for expansion of the membrane. This growth is continued until a flood breaches the membrane, causing the organism to be rapidly deteriorated by caustic agents.

In Figure 7.3, a single run persists at a constant level of population. However, this biomass is not caused by a stable population of cells but by another deformity.

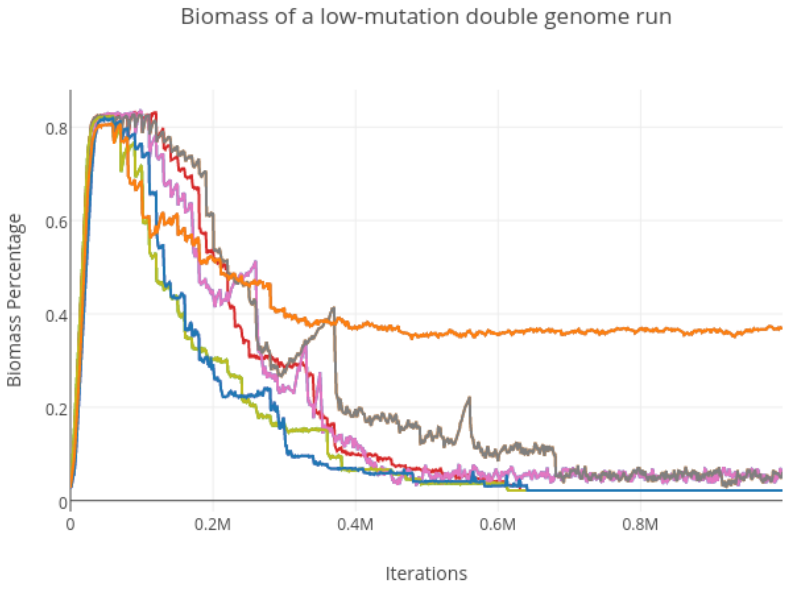

Figure 7.3: The results of three simulations with a mutation chance of .02. The genome of the organisms are modified by adding a second copy of each reaction. Each line represents a different simulation. The y-axis represents the percent biomass at different times during the simulation.

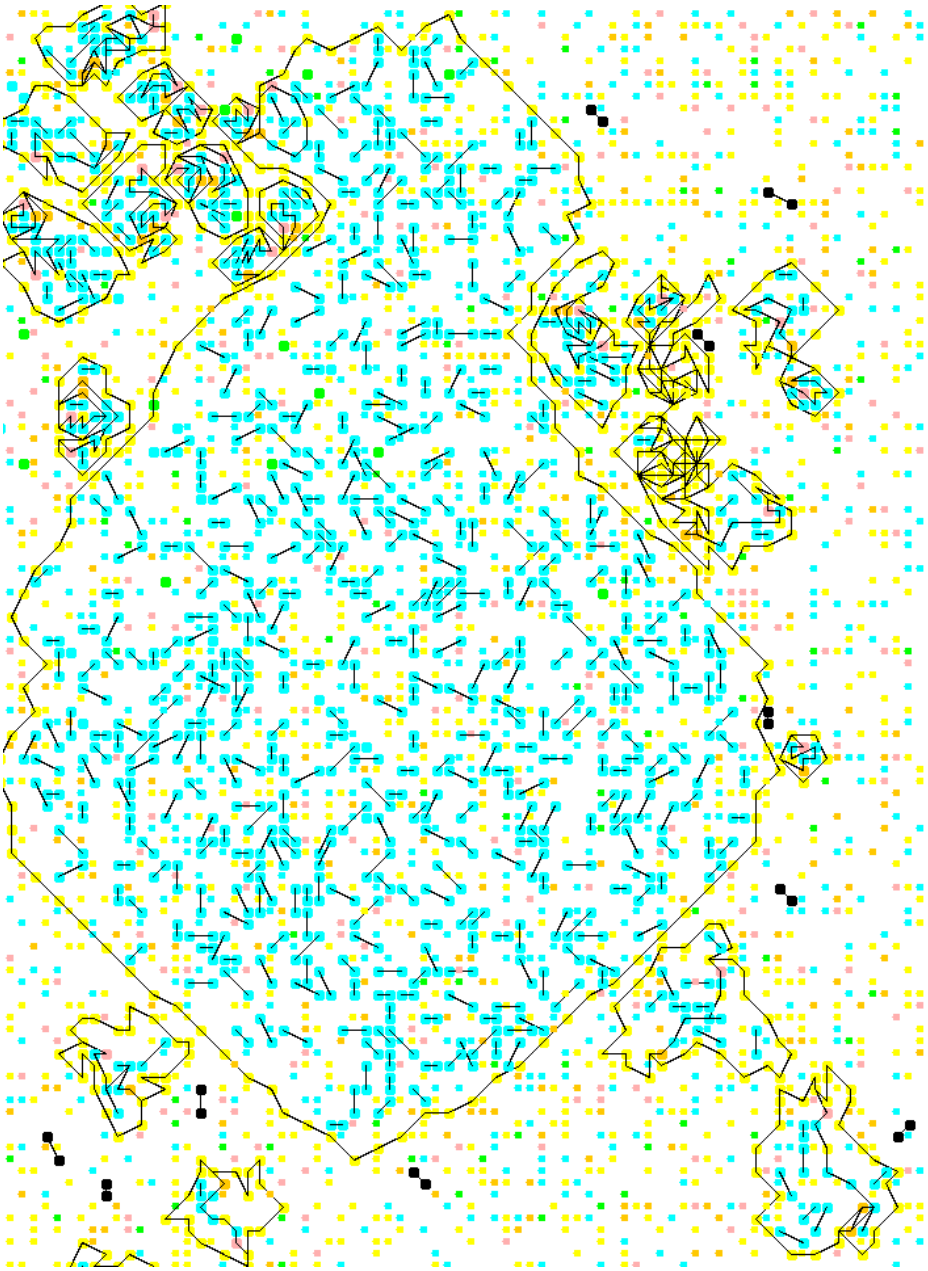

Figure 7.4: A single cell which has grown to a massive size. Blue represents type-B atoms, while yellow represents type-A atoms

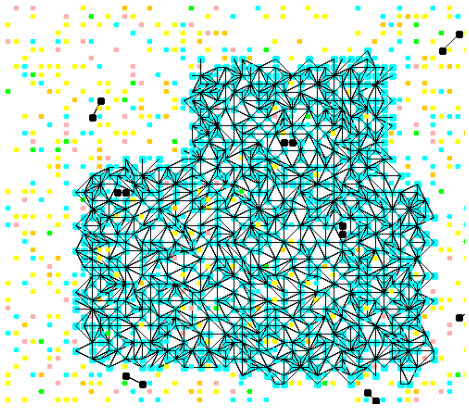

Figure 7.5: A malformed self-reproducing clump of enzymes

As seen in Figure 7.5, a merge mutation causes a self-propagating deformity. This clump of enzymes will absorb all free type-B atoms, copying the reaction data. The rapid propagation and structural stability makes these clumps resistant to the floods which the massive cells were vulnerable to.

In conclusion, this experiment failed to find either a sustainable population or adaptive mutations. However, interesting behaviors were observed as a result of our new mutation mechanisms, confirming that our mechanisms have the potential for more interesting behaviors than direct mutations.

7.2.4 Declining Biomass

In general, despite some abnormalities biomass continues to decline. However, hopeful behavior is observed more and more with this set of parameters. Consider the beginning of a single run.

As seen in Figure 7.6, biomass increases sharply until the population reaches maximum capacity. Once floods and mutations start, floods periodically lower the biomass. However, the population is still viable at this point, and reproducing organisms quickly absorb the free nutrients and return the biomass to its maximum capacity. This is, in fact, the exact pattern we would expect to see with a vital population. After the seventh flood, however, this pattern abruptly ceases. There is one inter-flood period in which slight growth is observed. After that no growth at all (aside from slight variations, which can be explained by membrane growth) can be observed.

As you can see in Figure 7.7, each line follows a similar pattern as that of Figure 7.6. Each individual line has a point beyond which biomass growth rapidly slows down and stops. Notably, these points are not all the same: great variation is observed in

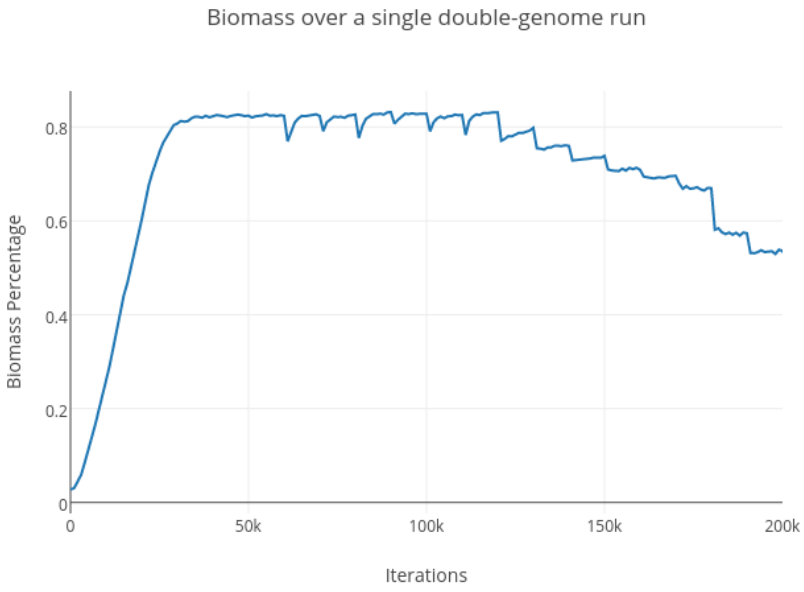

Figure 7.6: The first 200,000 iterations of a representative run from Figure 7.3, presented here for clarity

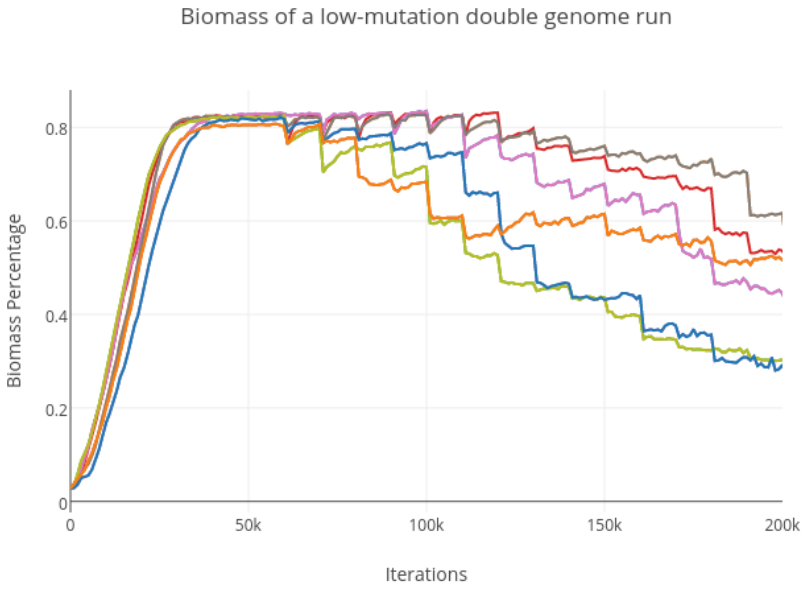

Figure 7.7: The first 200,000 iterations of Figure 7.3

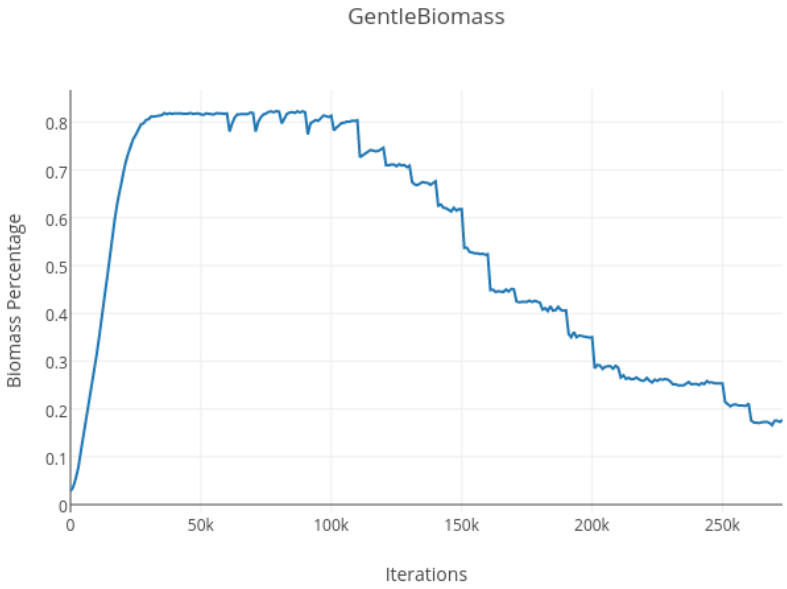

Figure 7.8: The first 200,000 iterations of a single run with a 1% mutation chance and a 10 cell flood radius.

the distribution of these points, though they all occur well before 150,000 iterations.

Finally, consider the results of a run with much gentler conditions. Mutation chance was reduced from 2% to 1%, and the radius of each flood was cut from 20 cells to 10 cells (reducing the total affected area by three-quarters).

In Figure 7.8, you can see that these changes have little, if any effect on the viability of the population. While actual decay of biomass takes place slightly slower, owing to the reduced flood area, the point of stagnant growth takes place at approximately the same time. It is unlikely, therefore, that any change to the parameters of the experiment can produce a viable population.

We still do not know why growth suddenly stops. The biomass data is consistent with a massively destructive enzyme. Such an enzyme, once mutated, would render all organisms permanently sterile. However, we cannot find any direct evidence of such an enzyme.

Chapter 8

Conclusion

We have proposed a great deal of modifications to Hutton's artificial chemistry system. While these modifications vary greatly in scope and effect, we can isolate a few driving principles behind our changes:

- We have tried to reduce the time needed for an organism to reproduce, so that we can simulate more generations with less computational time.
- We have tried to reduce hardcoded behavior to allow for a greater variety in potential organisms.
- We have tried to represent and mutate our genome in a way which would encourage more useful mutations.

In general, we expect that these three objectives have favorable effects on the strength of a simulation. However, we are unable to test these objectives. Our simulations function at the same level as Hutton's simulations: that is, no significant evolutionary action is observed. As success in this endeavor is generally binary, it is difficult to isolate any positive or negative effects of the changes we have made, as we do not have any ability to measure relative success. It is our hope, however, that the concepts behind our modifications can serve as jumping-off points for future systems.

8.1 Future Work

Several problems are still open for exploration. The most immediate is to address the technical problems encountered by our simulations, and to sustain a population of organisms reliably. More broadly, there are several other questions that remain unanswered. In particular, we would like a quantitative way to test the effects of each

modification. Such a method would likely require some numeric value which would capture a systems potential for evolution. Of course, the ultimate future achievement remains the development of a system which shows open-ended evolution.

Bibliography

[1] David H Ackley and Trent R Small. Indefinitely scalable computing= artificial life engineering. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 3–5, 2014.

[2] Matthew Cook. Universality in elementary cellular automata. *Complex Systems*, 15(1):1–40, 2004.

[3] Alan Dorin and Kevin B Korb. Building virtual ecosystems from artificial chemistry. In *Advances in Artificial Life*, pages 103–112. Springer, 2007.

[4] Adam Faulconbridge, Susan Stepney, Julian F Miller, and Leo SD Caves. Rbn-world. In *Advances in Artificial Life. Darwin Meets von Neumann*, pages 377–384. Springer, 2011.

[5] R.Wm. Gosper. Exploiting regularities in large cellular spaces. *Physica D: Nonlinear Phenomena*, 10(1-2):75–80, 1984.

[6] Margaret Hill, Susan Stepney, and Francis Wan. Penrose Life: ash and oscillators. *Advances in Artificial Life*, pages 471–480, 2005.

[7] Tim J Hutton. Replicators that make all their own rules. In *Proc. Workshop on Artif. Chem. and Its App., 8th European Conf. on Artif. Life*, 2005.

[8] Tim J Hutton. Evolvable self-reproducing cells in a two-dimensional artificial chemistry. *Artificial life*, 13(1):11–30, 2007.

[9] Tim J Hutton. The organic builder: a public experiment in artificial chemistries and self-replication. *Artificial life*, 15(1):21–8, 2009.

[10] Charles Ofria and Claus O Wilke. Avida: A software platform for research in computational evolutionary biology. *Artificial life*, 10(2):191–229, 2004.

[11] Alexandre Persat, Carey D Nadell, Minyoung Kevin Kim, Francois Ingremeau, Albert Siryaporn, Knut Drescher, Ned S Wingreen, Bonnie L Bassler, Zemer Gitai, and Howard A Stone. The mechanical world of bacteria. *Cell*, 161(5):988–997, 2015.

[12] Tomas G Rokicki. An algorithm for compressing space and time-amazing performance improvements can result. *DR DOBBS JOURNAL*, 31(4):12–+, 2006.

[13] Tim Taylor. Creativity in evolution: Individuals, interactions and environments. *Creative evolutionary systems*, pages 1–15, 2001.