Conservation of Matter Increases Evolutionary Activity

Simon Hickinbotham  Susan Stepney

Department of Computer Science, University of York, York YO10 5GH, UK,
sjh518@york.ac.uk

Abstract

We explore the hypothesis that adding conservation of matter to an artificial life system can increase its evolutionary activity, through experiments with the Stringmol artificial chemistry. Our first experiment examines the effect of varying the number of opcodes and finds a concentration which maximises the evolutionary activity of the system. The second experiment searches for the optimum relative concentrations of opcodes that maximises evolutionary activity: it finds increased evolutionary activity, a high diversity of opcode concentrations in each search run, and a different configuration of concentrations in separate search runs. The third experiment investigates the need for low concentrations of opcodes in high evolutionary activity, and finds that evo activity decreases when more of these particular opcodes are provided. We conclude that conservation of matter provides an important evolutionary pressure that can lead to more diversity and more evolutionary activity, and is therefore a desirable property for experiments in evolving ALife systems.

keywords: artificial chemistry; automata chemistry; evolutionary activity

Introduction: the role of matter in evolving systems

Biological evolutionary systems exist in the physical world and so they must conserve matter, yet virtual evolutionary systems are not obliged to conform to this constraint. What pressures does the conservation of matter place on an evolving system? Can an evolving system exploit this constraint? What advantages might it confer on synthetic evolutionary systems? Here we investigate this idea by examining the effect of imposing an additional constraint (limitation of material resources) on the ‘evolutionary activity’ generated by an automata chemistry.

We have previously demonstrated how a suitable artificial energy flux can lead to an evolutionary diversity “sweet spot”: a low flux is too constraining to allow exploration the evolutionary landscape; a high flux provides no incentive to do so (Hoverd and Stepney, 2011). Can conservation of artificial matter provide similar advantages? Schneider and Sagan (2005) make clear the distinction between open energy flux and closed material recycling in ecosystems. Lones et al. (2013) show an artificial system where an analogue of conservation of matter allows evolved solutions to be more readily found. Fernando and Rowe (2007) have experimented with conservation of matter in a simple artificial chemistry, and found that the restrictions it imposes on a reacting system can drive it to self-organise. However, these ideas have not previously been applied to the domain of automata chemistries ( Dittrich et al., 2001), in which the reactions between artificial molecules follows a program encoded in the structure of the molecule.

Stringmol (Hickinbotham et al., 2012) is an automata chemistry designed to explore intrinsic evolution in silico. It has demonstrated interesting behaviours such as hypercycles and emergent macro-mutations (Hickinbotham et al., 2010). In its basic form its execution rate is throttled by an externally imposed artificial energy flux, which provides selection pressure. Apart from in the work reported here, Stringmol is not constrained in how much “matter” it uses: its atomic opcodes are assumed to be available in arbitrarily large quantities.

Our central claim is that conservation of matter will increase evolutionary activity in an artificial system. We demonstrate this through three experiments. Our first experiment looks at the effect of having all opcodes being present in the same concentration, and searches for the absolute concentration that maximises evolutionary activity. Our second experiment searches for the optimum relative concentrations of opcodes that maximises evolutionary activity. Our third experiment takes evolved concentrations and determines whether high evolutionary activity is dependent upon low concentrations of key opcodes.

The structure of the rest of the paper is as follows. After providing some terminology, we provide some contextual material on conservation of matter in artificial chemistries (AChems). Next we discuss the modifications made to Stringmol to conserve matter. Then we summarise our measure of evolutionary activity used to quantify the differences made. We detail the three experiments, followed by a final section which discusses the results of our experiments in the
context of artificial life.

**Terminology**

It is necessary to give clear working definitions of some of the concepts we discuss at this point as many different definitions exist in the literature. Throughout the remainder of this paper, the following definitions are used:

- **Matter**: in biology, the basic unit of matter is the atom. By analogy, matter in a virtual system is anything that can be regarded as the basic unit of the composition of the system.

- **Conservation of Matter**: the basic principle that matter cannot be made or destroyed in a closed system.

- **Opcode**: the ‘atomic’ element of an automata chemistry. It is the building block of the sequence of the molecule. Each opcode has a function associated with it, allowing the sequence to be interpreted as a program.

- **Molecule**: any collection of one or more opcodes that has been assigned as a molecule by a Stringmol reaction. The difference between an opcode in ‘atomic form’ and a molecule with sequence length of 1 opcode is that the latter has the reactive structures (pointers and flags) that allow it to be executed as a program.

- **Concentration**: The total number of opcodes present in a simulation. (This definition is applicable to systems that have no concept of space.)

- **Species**: a type of molecule, identified by a unique sequence of opcodes.

- **Lifetime**: The number of timesteps in a simulation. The simulation ends either if there are no molecules remaining, or if a pre-specified number of timesteps are executed.

- **Cohort**: the set of molecules in an individual simulation at a given time.

- **Population**: the ensemble of simulations used in a genetic algorithm (note the distinction between ‘population’ and ‘cohort’).

**Conservation of matter in evolving systems**

Real world chemistry conserves matter: in a closed system, the atomic materials are always present throughout the time that the system is closed. Without conservation of matter, there is nothing wrong with a system performing an operation like \( X \rightarrow 2X \): the result is potentially unbounded growth. *With* conservation of matter, the reaction can only occur if there are sufficient raw materials in the system to make another instance of \( X \). Thus, growth is bounded.

AChems and artificial evolutionary systems tend to be implemented without the notion of conservation of matter (CoM). As in biological evolutionary systems, artificial evolutionary systems have to have a cohort of entities which exhibit variation in their forms. The variation is set by the initial cohorts and perpetuated through the process of mutation. Selection then operates on the heterogeneous cohort. Unlike in biological evolution, mutation is an event which happens stochastically, and at a pre-specified rate.

Our hypothesis is that conservation of matter will confer the following benefits on an evolving artificial system. Since matter cannot be created or destroyed, the problem of ‘bloat’ is solved immediately: the system cannot get larger than the resources available to it. In a similar manner, the issue of molecular parasites swamping the system is reduced, since they cannot undergo unbounded exponential growth: resource limits impose a carrying capacity. Conservation of matter could also be linked to mutation and emergence. Limited resources on replication of the genome could lead to errors on copy and temporary limitation of resources may make alternative strategies for survival relatively less costly.

**Quantifying the effect by Measuring Evolutionary Activity**

We are setting out to test the hypothesis that CoM has a positive influence on the evolutionary activity in a system. One problem with making this link is that the phenomenon of evolvability, also known as evolutionary activity, or the rate of evolution, is vague and difficult to measure quantitatively.

Here we recap the measure of evolutionary activity first presented in Droop and Hickinbotham (2012), which gives a numeric summary of evolutionary activity and thus allows different configurations to be compared.

The reasoning behind the measure is as follows. It is assumed that a species demonstrating some new beneficial adaptation will exhibit a rapidly increasing cohort size. By contrast, a neutrally-drifting species will not show any rapid increase in cohort size. Therefore by creating a measure of cohorts that are rapidly increasing, we can detect evolutionary activity.

Let \( c^i_t \) be the number of molecules of species \( i \) at timestep \( t \). The total cohort size at timestep \( t \) is:

\[
C_t = \sum_i c^i_t
\]  

(1)

The proportion of species \( i \) at timestep \( t \) is:

\[
p^i_t = c^i_t / C_t
\]  

(2)

The expected proportion of species \( i \) at timestep \( t \) is the proportion observed at the previous timestep:

\[
e^i_t = \begin{cases} p^i_{t-1} & \text{if } 0 < t \\ 0 & \text{if } t = 0 \end{cases}
\]  

(3)

The activity of species \( i \) at timestep \( t \) is defined to be the square of the excess of observed over expected proportion.

scaled by cohort size:

\[
a_i^t = \begin{cases} 
(p_i^t - e_i^t)^2 & \text{if } e_i^t < p_i^t \\
0 & \text{otherwise}
\end{cases}
\tag{4}
\]

This definition emphasises large positive increases in cohort size for each species, particularly where this occurs within a large cohort.

The total non-neutral activity \(A_Q\) of the simulation is the sum of each species activity at each timestep:

\[
A_Q = \sum_{t=0}^{T} \sum_{i} a_i^t
\tag{5}
\]

The measure \(A_Q\) has two advantages. Firstly, it is quantitative: it delivers a numerical measure which allows systems to be compared. Secondly, it measures non-neutral evolutionary activity without the need for an explicit model of neutral evolution. The measure is called Quantitative, non-neutral (QNN) evolutionary activity. Note that QNN is applicable to any systems where changes in cohort over time can be measured. An implementation of the measure in the R programming language is available from the first author’s website1.

**Methods**

**Experimental Vehicle: Stringmol Automata Chemistry**

In these experiments, we extended the original configuration of Stringmol described in Hickinbotham et al. (2010, 2012). Stringmol is a modern automata chemistry designed to be much simpler than its forebears by placing less emphasis on registers and memory addressing, and more emphasis on the process of binding as a precursor to a reaction between molecules. For full details of the Stringmol system see Hickinbotham et al. (2012). We give a brief description here. A molecule in Stringmol consists of a string of opcodes and four program pointers. There are no queues for processor time or death; both of these are allocated stochastically.

There are 33 opcodes. Most of these (the alphabetical characters) are ‘n-ops’: they have no function other than to form sequences of codes which are searched for by the functional codes. The seven functional codes (the non-alphabetical characters) manipulate the position of pointers, using inexact sequence matching to determine program flow.

Molecules run their programs only when they bind to other molecules, as shown in figure 1, meaning that a molecule has no opportunity to interfere with its neighbours, unless it can bind to it. This was designed to emulate the specific binding properties of enzymes and substrates, and makes the system much less noisy than other automata chemistries such as Tierra.

1see http://www-users.cs.york.ac.uk/~sjh/software

---

Molecules survive and multiply by being created more quickly (on average) than they are destroyed. The creation of molecules usually occurs by a process of copying. Figure 1 illustrates a reaction between two molecules after they have bound. The first half of this molecule specifies the binding rate and the second half formulates the program that creates a copy of the molecule. The reaction is initialised by determining which molecule will execute its sequence. The instruction pointer (\(1\) in figure 1) of the executing molecule ‘r’ steps through the sequence of opcodes, executing each in turn. It reaches a part of the sequence that executes an iterative copy of the partner molecule ‘m’. This is achieved by; copying the opcode at the read pointer \(R\); to the write pointer \(W\); incrementing \(R\) and \(W\); checking the position of \(R\), and moving \(1\) to the flow pointer \(F\) if the partner molecule is not fully copied. After this has happened, \(1\) exits the loop, and executes another sequence of instructions that turns the newly-created sequence of opcodes into a new molecule. Cohorts of molecules containing variants of this program that successfully create new copies of each other sufficiently quickly are able to maintain themselves indefinitely.

**Conserving Matter in Automata Chemistries**

To experiment with conservation of matter, we extend the standard Stringmol framework detailed in Hickinbotham et al. (2012) to conserve matter, to produce CoM-Stringmol. The changes are as follows:

- **Matter** is introduced explicitly by setting a fixed number of instances of each opcode in the system, and keeping a record of how many opcodes of each type are “free”, i.e. not forming part of a string molecule. We refer to this record as the opcode buffer. This is an array of integers in Stringmol, but it is analogous to the atoms existing in solution in a physical system.

- **Operators** are modified so that it is not possible to change the total number of instances of an opcode. Only the copy operator ‘⇐’ needs to be changed in order to achieve this. This operator’s function is to insert an opcode at the write pointer (shown as \(W\) in figure 1), which overwrites any opcode at that location on the string molecule unless the write pointer is at the end of a string. The changes needed in order to conserve matter are straightforward: decrement the
Seed Molecule

Each run commences with 150 instances of the ‘seed’ molecule, the sequence of which is shown in figure 2. This is similar to the ‘replicase’ molecule used in Hickinbotham et al. (2010), but with a more balanced distribution of the n-op code letters (see figure 3). The function of the molecule is identical to the earlier version. In the CoM formulation of Stringmol, replication is exact unless there is a shortage of available opcodes in the opcode buffer.

<table>
<thead>
<tr>
<th>opcode</th>
<th>count</th>
<th>opcode</th>
<th>count</th>
<th>opcode</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>6</td>
<td>H</td>
<td>5</td>
<td>&gt;</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>~</td>
<td>1</td>
<td>R</td>
<td>5</td>
</tr>
<tr>
<td>$</td>
<td>4</td>
<td>J</td>
<td>5</td>
<td>=</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>1</td>
<td>?</td>
<td>1</td>
<td>U</td>
<td>7</td>
</tr>
<tr>
<td>E</td>
<td>3</td>
<td>L</td>
<td>5</td>
<td>W</td>
<td>6</td>
</tr>
<tr>
<td>%</td>
<td>1</td>
<td>}</td>
<td>1</td>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>G</td>
<td>2</td>
<td>O</td>
<td>4</td>
<td>Y</td>
<td>2</td>
</tr>
</tbody>
</table>

Figure 3: Number of opcodes used in each of the 150 seed molecules. One seed molecule uses a total of 64 opcodes.

Experiments

We now detail three experiments that characterise the interaction between CoM-Stringmol and evolutionary activity. The first experiment presents a series of evaluations, each with the same fixed concentration for every opcode in the Stringmol chemistry. The second uses an evolutionary search algorithm to find a set of concentrations that produce higher levels of evolutionary activity, by varying the individual concentrations of each opcode. The third experiment examines the effect of low concentrations of opcodes on the amount of evolutionary activity.

Experiment 1: Exploring equal concentrations of opcodes

We set the total concentration of each opcode present in the system to a constant $\lambda$. We ran a set of configurations varying $\lambda$ between 1,000 and 3,000 opcodes per container. For example, if we set $\lambda = 1,000$, then since there are 33 op-code types, there would be a total of 33,000 opcodes in the container. For each value of $\lambda$, we performed 100 runs of Stringmol using the seed molecule described above. This allows us to benchmark the sensitivity of the QNN measure and evaluate its performance with respect to detecting the effects of CoM.

Our hypothesis is that high opcode concentrations would mean low evolutionary activity since there would be no substitutions or deletions. Low opcode concentrations would mean no functional replicate molecules could be built anew, so the cohort of molecules would be unable to self-maintain, again resulting in little evolutionary activity. Between these extremes, we predict that a “sweet spot” exists, where the concentrations allowed a self-maintaining cohort to exist, but which also induced sufficient opcode substitutions for a higher rate of evolutionary activity to emerge.

We gathered the following statistics for each run: lifetime of the simulation; number of molecular species created; QNN activity. Figure 4 shows the results of these runs.
There are several points to note:

1. The Lifetimes of runs rarely reach the limit of one million timesteps since the population of molecules cannot maintain itself under these conditions. Early extinction is not uncommon in Stringmol without CoM, where the runs would continue indefinitely without mutation. With CoM, runs are particularly short where $\lambda < 2400$.

2. The number of species produced peaks at $\lambda = 1700$. However, the lifetime of runs is very short at the concentration where the number of species is at its maximum. The number of species created could be interpreted as an indicator of evolutionary activity. This exposes a problem with using the number of species as a direct measure of evolutionary activity: systems which generate diversity do not necessarily have the ability to self-maintain. This is the “error catastrophe” of Eigen and Schuster (1977).

3. The QNN as a measure peaks at opcode concentration $\lambda = 2400$. This is the concentration above which the molecular cohort can maintain itself, as indicated by the lifetime of the simulations. The fact that the highest value of the QNN measure is at the boundary between error catastrophe and self-maintaining cohorts indicates that it is a useful measure of evolutionary activity.

In conclusion, this experiment shows that conserving matter in AChem systems not only has an effect on the ability of the system to self-maintain, but also has an ability to affect the rate of evolutionary activity as measured by QNN.

**Experiment II: Tuning Opcode Concentration**

Apart from expediency, there is no reason why $\lambda$ should be equal for all the opcodes in a system, just as different minerals are present in different concentrations in the physical world. In this experiment, we use an evolutionary search algorithm to find opcode concentrations that yield high evolutionary activity as measured by QNN.

We hypothesise that evolutionary activity could be improved if the concentrations of different opcodes could be different from each other.

To perform the search, we use the tournament-based microbial GA (Harvey, 2009). The mutation rate was set to 0.1, which is a high value for a genetic algorithm, but it seemed better to have a high value that explored the space of concentrations rather than a low value that might become trapped in a local optimum. There are 33 opcodes in the Stringmol system, so are 33 parameters in the genome encoding the concentration of each opcode. The concentration of each opcode was allowed to vary independently between 0 and 4000, encompassing the range of $\lambda$ values in Experiment I. Each evaluation required a complete run of Stringmol up to a maximum of 1 million timesteps. Even with a cluster of computers available to us, it was necessary to try to constrain the processing overhead to yield results in reasonable time. Accordingly, we limited the GA population size to 20, and used a single evaluation for each fitness measure, even though this was likely to introduce significant stochasticity into the QNN measure (as apparent from each bar-plot in figure 4). By constraining the processing in this manner, we were able to carry out 5000 tournaments during each of 25 GA runs.

**Results: Fitness changes in GA run**

We analysed the outputs of the 25 GA runs, and present a summary of the findings here. The left panel of figure 5 illustrates the change in fitness over the 5000 tournaments of three sample runs of the search algorithm. There were two distinct groups: one in which QNN fitness stayed below 200, and one in which a marked increase in fitness occurred at some point in the simulation. 5 of the 25 runs belong to the former group. The principal reason for this distribution can be seen in the right panel of figure 5.

The 20 highest-scoring runs all evolved concentrations of opcodes that allowed the cohort of molecules to self-maintain, as demonstrated by the lifetime reaching the limit set in the simulation. Here, early evaluations in the search went extinct quickly and had with low QNN values, indicating that low concentrations of key opcodes prevent the system from self-maintaining. The evolutionary search then found a set of concentrations that allowed the molecules to self-maintain, and QNN values increased. This is a side effect of using QNN as a fitness measure, as longer simulations have greater opportunity to innovate. In other words,
the QNN-based fitness measure places an indirect requirement on the system to be capable of self-maintaining.

Other important features of the trajectory of the search are:

1. The stochastic nature of individual Stringmol simulations result in variable fitness values for each configuration. However the search algorithm is sufficiently robust to find fit configurations of the opcode concentration levels.

2. Although the median fitness of the simulations increases by several orders of magnitude, individual simulations occasionally score poorly, again due to the stochasticity of the simulation.

3. After the search converges, the median fitness values regularly reach around 300, whereas the median fitness value for a uniform opcode concentration of 2400 was less than 100, leading us to conclude that the search algorithm yielded concentrations of opcodes that improved evolutionary activity.

Results: Opcode Concentrations

Figure 6 shows the distribution of opcode concentrations in the final population of the GA runs in figure 5. Although there are no common features in terms of individual opcode concentrations, we find that in each run the concentration of several opcodes is low.

From figure 7 we see that the concentration of the copy operator ‘=’ (lying between the ‘T’ and ‘U’ n-ops on the horizontal axis of figure 6) is often (but not always) low. The copy operator ‘=’ has key functionality in the self-replicating system: it specifies that the opcode at the read pointer should be inserted at the location of the write pointer. How can low concentrations of this operator lead to self-maintaining runs with high evolutionary activity? Experiment III addresses this question.

Experiment III: Boosting Concentrations

The previous experiment demonstrates that evolved concentrations of opcodes can yield higher evolutionary activity than setting all opcodes to the same concentration. Some opcodes that are essential for the self-maintaining system evolve to low concentrations in high-scoring runs of the GA. Our definition of ‘low’ concentration is any concentration of less than 400, which is 10% of the maximum permissible concentration in the GA.

The highest scoring run of the 25 we carried out in the previous system had low concentrations of the operators ‘D’, ‘E’, ‘=’ and ‘X’ (see figure 6 and figure 7). Each of these has a different function in the replicase molecule used in our experiments. The ‘D’ and ‘E’ are both used to form the part of the sequence that specifies the binding between molecules. The ‘X’ opcode is also present in the binding region but acts as an indel, serving to reduce the chance of molecules bind-
The ‘=’ opcode is central to the copy operation that allows molecules to create copies of other molecules.

We hypothesise that these opcode concentrations need to be low to promote evolutionary activity. To test this, we created a simulation where the four opcodes listed above were present in concentrations at 10 times the evolved value. We call this the ‘boosted’ system. We then evaluated 20 runs of the evolved system and the boosted system, and measured QNN for each run.

**Results** Figure 8 shows the distribution of the fittest evolved configuration from experiment I (‘fixed’), experiment II (‘fittest’) and the ‘boosted’ configuration. We have shown the distributions as boxplots overlaid with beanplots (Kampstra, 2008) to highlight the multi-modal distribution of QNN for the fittest evolved configuration.

Our interpretation of this data is as follows. The ‘boosted’ configuration is organised into a self-reproducing system that is robust, but demonstrates little evolutionary activity as detected by QNN. By contrast, the fittest configuration sets up dynamics that yield high-scoring evolutionary activity, but those dynamics also ‘risk’ extinction. The configuration evolved because in a population of replicating systems, this risk of extinction is ameliorated: there is usually a ‘sister’ system in the population that is doing well, and preserving the configuration in the population. Both of these configurations demonstrate more evolutionary activity than the best ‘fixed’ configuration from experiment I, demonstrating that varying the concentrations of opcodes can contribute to the overall innovation of the system.

**Discussion**

We have demonstrated that conservation of matter can be implemented in an existing ALife system.

In contrast to building a system anew, this approach allows us to explore the pros and cons of conservation of matter, and elucidate the differences between environments where resources are either conserved or available in unlimited quantities. This is important for virtual systems where extra effort is required to ensure that the system conserves matter: we need to establish whether the reward is worth the effort. It is also important for understanding the role of resource availability in physical systems, for example in understanding its role in the origin of life, and in developing self-assembling robotic systems. In our experiments, tuning the concentrations of opcodes to maximise QNN allows us to create CoM-Stringmol runs that innovate and perpetuate, possibly even more than the original Stringmol formulation.

Figure 8: Beanplots (in grey) overlaid with individual data points (in red) and boxplots for distributions of QNN values in three configurations of CoM-Stringmol. **Left:** $\lambda = 2400$ from Experiment I, **Middle:** fittest cohort found in Experiment II, and **Right:** ‘boosted’ concentrations of scarce op-codes from experiment III.

This contribution is also the first active use of the QNN measure, which was used both to measure a set of values and to drive a search algorithm. As we have seen, the role of the key operators in the Stringmol system can be evaluated at the cohort level, rather than at the level of an individual reaction.

CoM can be thought of as an embodied mutation operator that responds to local conditions. Where the molecular cohort is a good fit to its environment, mutation is low and the system self-maintains. Where the molecular cohort over-exploits resources, mutation increases and new configurations of molecules are found. This is the sort of feature that an evolving system will need in order to better exploit some of the stochastic programming features of Stringmol (see Hickinbotham et al. (2012)), rather than existing in a design space that is adjacent to hand-designed molecules.

**Future work:** We will attempt to determine what the link is between specific opcode concentrations and high-scoring runs, and how they self-maintain in situations where the concentration of key opcodes is so low. We also intend to apply the concept of CoM to Tierra and Avida, and to richer Stringmol systems.

The QNN measure of evolutionary activity allows us to use conservation of matter to meaningfully explore the local functional molecular space. However, we recognise that the measure should be part of a suite of analysis tools, that should capture information about increasing complexity and diversity in an evolving system.

One of the opportunities this work presents us with is that it allows us to generate different levels of abstraction, and so build more abstract simulations that are correct versions of low level systems like Stringmol. It is important that virtual systems conserve matter if such dual representations are needed (Nellis and Stepney, 2010).

Finally we will use CoM to study the effect chemical flux, to see if systems can adapt to changing ratios of opcodes (Pascal et al., 2013).

**Acknowledgements**

The experiments were performed using the Volvox cluster at the York Centre for Complex Systems Analysis. Part of this work was funded by the EU FP7 project EvoEvo, grant number 610427.

**References**


