Degeneracy Enriches Artificial Chemistry Binding Systems

Ed Clark[1], Adam Nellis[1], Simon Hickinbotham[1], Susan Stepney[1], Tim Clarke[1], Mungo Pay[1], Peter Young[1]
[1]York Centre for Complex Systems Analysis, University of York, UK, Y010 5GE
edclark@cs.york.ac.uk

Abstract

We hypothesise that degeneracy in the components of an artificial chemistry (AChem) facilitates the complexity of the system as a whole. We introduce definitions of degeneracy and redundancy, and show how these quantities can be calculated for the binding system of an AChem.

We present a case study using the AChem Stringmol, in order to support our hypothesis. We demonstrate that the binding system in Stringmol has degeneracy and we create a deliberately poor variant: 'sticky-Stringmol', that has a binding system with no degeneracy. Comparing sticky-Stringmol to Stringmol, we note the loss of many simulation artifacts that have been used as evidence of the complexity of Stringmol, including: emergent macro-mutations, hypercycles, sweeps and parasite evasion. These results are evidence that degeneracy in the components of an AChem facilitates the complexity of the system as a whole.

Introduction

Degeneracy is the ability of elements that are different, in some respect, to perform the same role in some, but not all, situations. Degeneracy is a noticeable property of many biological systems, and is observable on many scales within those systems [7] and has been linked to the evolvability and robustness of these systems [16]. Examples range from molecular interactions and gene networks [7], the connectivity of neurons in the brain [13], through to social networks [15]. Complexity and degeneracy have been strongly linked [14]. Attempts have been made to describe these concepts into mathematically meaningful, and consequently unambiguous formulae [14][7].

Just as degeneracy can be observed on many scales in nature, so it should be in artificial chemistries (AChems) that aspire to achieve the levels of complexity that exist in the natural world. We hypothesise that degeneracy in the components of an AChem will facilitate complexity of the system as a whole. We introduce measures of degeneracy and redundancy in terms of an 'interaction function' between two sets. We use binding between two sets of chemicals in an AChem (defined below) as a concrete example of an interaction function. We demonstrate that the degeneracy in the binding system of Stringmol [10] is particularly important for the complexity of the AChem as a whole.

When presenting the complexity of an AChem, it is standard practice to present simulation results and focus on an artifact that the system has been able to produce as evidence of the complexity of the AChem. Examples of artifacts include: the ability to 'compute' prime numbers [1]; the generation of cooperative organisations [9], hypercycles [10] and autocatalytic sets [12]. However, the complexity available in current AChems is still well below that of the natural world.

The presentation of simulation artifacts is currently the only available way to evaluate AChems (see [4]). As such, two chemistries that produce different types of artifact can only be compared in a qualitative manner. Progress has been made on formalising artifacts in chemistries, and automating the discovery of autocatalytic sets [12] and organisations [5]. However, simulation artifacts can only be measured *a posteriori*: they can not be determined at design stage. The degeneracy measure we introduce can be applied at the design stage to the components of an AChem, thus allowing sources of complexity to be designed in.

Binding in AChems

In this paper, we focus on degeneracy in the context of binding in AChems. In the '(S, R, A)' definition of AChems [4], S is a set of chemicals, R is a set of reactions between the chemicals and A is the algorithm that applies reactions from R to chemicals from S. For example, if the chemicals in set S are integers, then the set R of reactions might contain all reactions of the form:

$$a + b \mapsto c \qquad \text{if } c = \tfrac{a}{b} \text{ is an integer.} \tag{1}$$

This is the prime number generation chemistry [1].

The important point for this discussion is the binding rule: "if $\frac{a}{b}$ is an integer". This can be viewed as an "*if-then*" statement: *if* the binding rule is true, the reaction may proceed. The left hand side (LHS) of the reaction, "$a + b$", is the *if* part of this statement. The right hand side (RHS) of the reaction, "$\mapsto c$", is the *then* part. Looking at the chemistries reviewed in [4], the vast majority have a trivial LHS, where the *if* simply tests if two molecules are presented by the algorithm, A. The only AChems we are aware of with a non-trivial LHS are Primes [1], AlChemy [9], Stringmol [10], Molecular Classifier Systems [3] and RBN-world [8].

AlChemy (level 0) had relatively simple binding, which resulted in the collapse of the system into 'self-replicators'. In AlChemy (level 1), binds that would result in reactions that propagate self replicators were restricted. As a result of the enriched binding rule, AlChemy (level 1) produced more complex artifacts, including 'cooperative organisations' [9]. This example helps support our hypothesis that binding is an important component of an AChem, and that changes to this component can change the level of complexity observed in the system.

Organisation of the Paper

We define degeneracy and redundancy in an unambiguous manner, and introduce methods to measure these quantities. We justify introducing a new measure of degeneracy instead of adopting previously published measures. We use our measures to analyse the binding system used in Stringmol and demonstrate that the binding system is is capable of producing degeneracy. We also use these measures to demonstrate that ubiquitous binding is unable to produce degeneracy. We use ubiquitous binding to define a deliberately poor Stringmol variant: 'sticky-Stringmol'. We replicate the experimental procedures of [10] in order to compare the artifacts of 'sticky-Stringmol' and Stringmol. We give an overview of the previously undetected phenomena of 'parasite evasion' in Stringmol containers. The two mechanisms by which the container is able to survive a potentially fatal parasite are linked to binding. We also find that sticky-Sringmol containers are unable to evade a parasite.

Degeneracy and Redundancy

We formally introduce and define redundancy and degeneracy in abstract terms, and provide a worked example calculating the redundancy and degeneracy of the binding system of a fictitious AChem.

In order to make an unambiguous statement of redundancy or degeneracy, one must state three pieces of information: Two sets of elements, A and B, that are being compared, and the method of comparison, defined by an 'interaction function', $f : A \times B \mapsto \{0, 1\}$, stating whether an element of A and an element of B interact or not.

If we consider an arbitrary element, a_m of set A, we can define a subset $B_{a_m} \subseteq B$, in terms of (a_m, f, B), containing all the elements of B that a_m interacts with:

$$B_{a_m} = \{b \in B \mid f(a_m, b) = 1\}. \tag{2}$$

Elements a_m and a_n, are *redundant* if

$$\mathcal{R}(a_m, a_n \mid f, B) \Leftrightarrow B_{a_m} = B_{a_n}. \tag{3}$$

Elements a_m and a_n, are *degenerate* if

$$\mathcal{D}(a_m, a_n \mid f, B) \Leftrightarrow (B_{a_m} \neq B_{a_n} \;\&\; B_{a_m} \cap B_{a_n} \neq \emptyset). \tag{4}$$

The definitions in equations 3 and 4 equip us to deal with questions concerning individual examples such as 'are a_1 and a_2 degenerate or redundant in a given context'. The ability to determine if $\mathcal{D}(a_1, a_2|f, B)$ is true, does not equip us to answer more general questions, such as what is the degeneracy *of a set* in a given context, $\mathcal{D}(A|f, B)$.

Degeneracy and redundancy, even when clearly defined between elements, have a non-trivial interaction within a set. Consider: sets $C, D = \{a, b, c, d, f, g\}$, and some interaction function f that causes the resulting matrix, which can be viewed as a network, to contain examples of both degeneracy and redundancy, see figure 1a. Consider also $A, B = \{a, b, c, d, e, f, g\}$, where e is part of a redundant set with a, see figure 1b. True measures of degeneracy and redundancy should detect that the redundancy of the set C is different from the set A. However, is the degeneracy of set C the same as the degeneracy of set A? If one wishes to maintain degeneracy of a set and redundancy of the set as orthogonal concepts, then the answer to this question must be 'no'. If one answers 'yes', then the concept of the degeneracy of a set becomes conflated with the redundancy of the set. As a result of this conflation, such measures of degeneracy lose their value, as the results they give may be skewed by redundancy. This is why we introduce a new measure of degeneracy, rather than adopting an existing measure. The key to understanding the relationship between degeneracy and redundancy, is knowing that it is possible to measure the redundancy of a set without regard for the degeneracy of a set, but not the other way around.

It is, however, possible to construct a measure of degeneracy of a set that does not suffer from this conflation with redundancy, keeping the mathematical concepts of degeneracy and redundancy of sets orthogonal. We introduce such a measure here. Our method avoids the conflation problem by accounting for the redundancy of the two sets (in the contexts of an interaction function f) and constructing new sets that have no redundancy. The set $\hat{A}$ is constructed from the set A (in the context of set B and the interaction function f) such that the elements $\hat{a} \in \hat{A}$ are the redundant sets of A. We can construct $\hat{B}$ in a similar manner. Note that it makes no difference if we construct $\hat{B}$ in the context of $\hat{A}$ or the context of A. These constructions can take place in any order and all examples of degeneracy that exist in (A, B) are maintained in $(\hat{A}, \hat{B})$.

Each element $\hat{a}$ of the reduced set $\hat{A}$ is itself a set containing one or more redundant elements from A. It is on these redundant sets that we base our measure of degeneracy. If we reconsider the above thought experiment, it can be seen that the element e will join an existing redundant set, see figure 1 parts (c) and (d). Consequently it will not

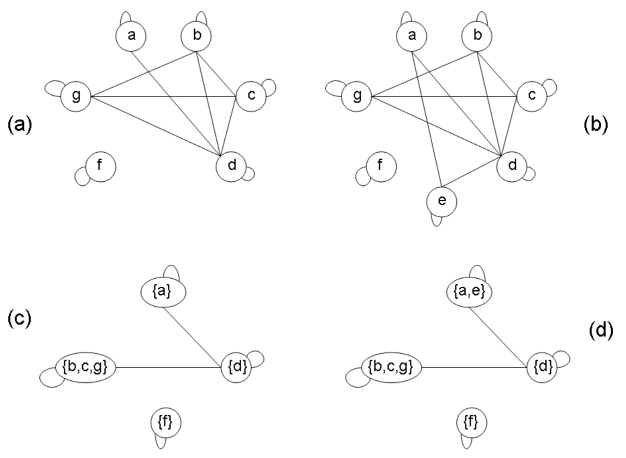


Figure 1: A network example of interaction between elements of a set. Two nodes x and y are joined with an edge if $f(x, y) = 1$. (a) is an example of an interaction containing examples of both degeneracy and redundancy by the definitions given in equations 3 and 4. In (b) the element 'e' has been added. The elements a and e form a redundant set $\{a, e\}$ as they both bind to elements $\{a, e, d\}$. (c) and (d), show the same relationships as (a) and (b) respectively, but in terms of redundant sets rather than elements.

affect a measure of degeneracy that is based on the elements of $\hat{A}$ (the redundant sets of the elements of A), instead of the elements of A.

We follow the definitions of redundancy and degeneracy for pairs of elements, and define redundancy and degeneracy for sets. Firstly, we consider an arbitrary element of set $\hat{A}$, $\hat{a}$, and define the subset $\hat{B}_{\hat{a}_m} \subseteq \hat{B}$. This contains all the redundant sets of $\hat{B}$ that $\hat{a}$ interacts with:

$$\hat{B}_{\hat{a}_m} = \{\hat{b} \in \hat{B} \mid f(\hat{a}_m, \hat{b}) = 1\}. \quad (5)$$

We define the *redundancy of the set* A, in the context of (f, B), as the set of sizes of redundant sets of A:

$$\mathcal{R}(A \mid f, B) = \{|\hat{a}| \mid \hat{a} \in \hat{A}\}. \quad (6)$$

$\mathcal{R}(A|f, B)$ takes the form of a set of size $|\hat{A}|$; the elements of this set are the sizes of the sets $\hat{a} \in \hat{A}$.

We define the *degeneracy of the set* A in the context of (f, B):

$$\mathcal{D}(A \mid f, B) = \{|\hat{B}_{\hat{a}}| \mid \hat{a} \in \hat{A}\}. \quad (7)$$

$\mathcal{D}(A|f, B)$ also takes the form of a set of size $|\hat{A}|$; the elements of this set are the numbers of redundant sets in $\hat{B}$, that each element $\hat{a} \in \hat{A}$ interacts with.

Worked Example

We define $A, B = \{a, b, c, d, e, f, g\}$ to be all the chemicals in our fictitious chemistry. Note that for the purposes of this example, we do not need to specify the reaction rule, as the products of reactions do not concern us in this calculation. We assume the binding rule returns a probability; we can apply a threshold at zero in order to construct an interaction function f. The result of the thresholding is shown in figure 2a. As it contains only binary values, it is an interaction matrix and the definitions of degeneracy and redundancy given in equations 3 and 4 apply (as in figure 1b).

We now construct the redundant sets: In figure 2a the row a and the row e have the same values, as such they are redundant under the definition given in equation 3. Similarly, rows b, c and g all have the same values. We can construct the redundant sets $\hat{A} = \{\{a, e\}, \{b, c, g\}, \{d\}, \{f\}\}$; if we apply the same process to the columns, we obtain the reduced matrix shown in figure 2b (as in figure 1d).

The sizes of the redundant sets, shown in the row labels in figure 2b, make up the redundancy set, $\mathcal{R}(A \mid f, B) = \{2, 3, 1, 1\}$, shown in figure 2c. In order to quantitatively compare binding systems from different chemistries of different sizes we scale the redundancy set by dividing the values in the set by the average redundancy. The average redundancy is given by the sum of the set sizes divided by the number of sets; in this case (2+3+1+1)/4 = 7/4. The scaled redundancy set is the redundancy set divided by the average redundancy, shown in figure 2c.

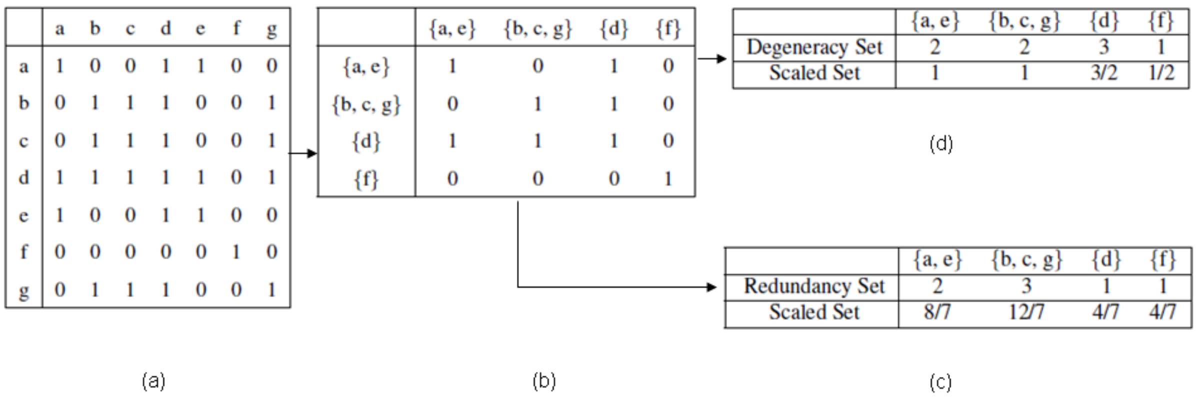
From the reduced interaction matrix in figure 2b, it is also possible to calculate the degeneracy set. The degeneracy of set A is obtained by summing the respective rows in the reduced interaction matrix in figure 2b, the result, $\mathcal{D}(A \mid f, B) = \{2, 2, 3, 1\}$, is shown in figure 2d. The degeneracy of set B would be obtained by summing the columns. Note that the calculation of degeneracy is not based on the elements of set A, but is instead based on $\hat{A}$, the redundant sets of A.

We rescale the degeneracy set by dividing the degeneracy set by the average degeneracy. The average degeneracy is calculated by summing all the elements in the interaction matrix in figure 2b and dividing that by the number of rows, $|\hat{A}|$. In this case the average degeneracy is 8/4=2. The scaled degeneracy set is shown in figure 2d.

These scaled sets can be used to compare the spread of redundancy and degeneracy when the systems being compared are of different sizes. A scatter plot is ideal for such a comparison, the rescaled degeneracy and redundancy sets from the worked example are shown in figure 3. If the systems being compared are the same size then it is appropriate to used the unscaled sets, allowing comparison of both the relative spread and the actual values of redundancy and degeneracy.

Results

We apply our measures of degeneracy and redundancy (defined in equations 6 and 7) to the binding system used in Stringmol and to 'ubiquitous binding' (all molecules bind). This makes ubiquitous binding a good candidate for testing



	a	b	c	d	e	f	g
a	1	0	0	1	1	0	0
b	0	1	1	1	0	0	1
c	0	1	1	1	0	0	1
d	1	1	1	1	1	0	1
e	1	0	0	1	1	0	0
f	0	0	0	0	0	1	0
g	0	1	1	1	0	0	1

(a)

	{a, e}	{b, c, g}	{d}	{f}
{a, e}	1	0	1	0
{b, c, g}	0	1	1	0
{d}	1	1	1	0
{f}	0	0	0	1

(b)

	{a, e}	{b, c, g}	{d}	{f}
Redundancy Set	2	3	1	1
Scaled Set	8/7	12/7	4/7	4/7

(c)

	{a, e}	{b, c, g}	{d}	{f}
Degeneracy Set	2	2	3	1
Scaled Set	1	1	3/2	1/2

(d)

Figure 2: (a): The interaction matrix: $\{a, b, c, d, e, f, g\}$ are the chemicals of set A, 1 indicating two molecules bind and 0 indicating two molecules do not bind. (b), the reduced matrix: The elements of $\hat{A}$ are the redundant sets of A, these sets are given explicitly as the row and column labels. (c): The redundancy set and scaled redundancy set, for set A. The values of the redundancy set are the number of elements in row labels of (b). The scaled redundancy set is obtained by dividing the unscaled set by the average redundancy. (d): The degeneracy set and scaled degeneracy set for set A. The values of the degeneracy set are the number of ones on each row in the reduced matrix (b). The scaled degeneracy set is obtained by dividing through by the average degeneracy in $\hat{A}$.

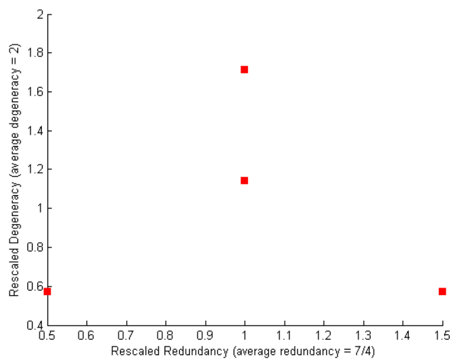


Figure 3: Redundancy and degeneracy scatter plot for set A in the worked example. Each point in the scatter represents a redundant set (an element of $\hat{A}$). The position on the redundancy axis is given by the scaled redundancy set shown in figure 2(c). Similarly, the position of the degeneracy axis is given by the the scaled degeneracy set shown in figure 2(d). The redundancy and degeneracy sets are scaled such that the center of mass of the scatter plot is at $(1, 1)$.

our hypothesis: that a more simplistic approach to binding will negatively impact the complexity of the artifacts observed in simulations. We apply the methodology of [10] to sticky-Stringmol and compare our results. We then describe how mid run parasites are evaded in Stringmol and how the mechanism for evasion is lost in sticky-Stringmol.

In our general comparison with Stringmol, as well as in the parasite trial, sticky-Stringmol is effectively the control experiment. By having a deliberately poor variant of Stringmol, we are able to establish which simulation artifacts are dependent on the degeneracy of the binding rule.

Measuring degeneracy

The Stringmol alphabet is 33 characters: 7 'functional' characters {$>^?=}%}, and 26 'non-functional' characters $\{A - Z\}$. Functional characters in Stringmol can contribute towards a bind site, but they contribute half as much as non-functional characters (for the full details, see [11]). We present results for the reduced character set: $\{AB\%CD\}$, containing 1 functional character and 4 non-functional characters, as the calculation for the full character set is intractable. We use the tailored Smith-Waterman algorithm [11] to calculate the bind strength of all strings of length 6 from this alphabet, and threshold at a Smith-Waterman score of 0.75 to produce an interaction function.

Degeneracy and redundancy for the set of all strings of length 6 are shown in figure 4. We also present the degeneracy and redundancy for the ubiquitous binding system used in sticky-Stringmol (under the same conditions) on the same figure, to allow a direct comparison of the properties of the two binding systems. We argue only that the tailored Smith-Waterman algorithm is capable of producing degeneracy and redundancy, not the specific levels of this which can be achieved for a string of arbitrary length. For ubiquitous binding, the matrix of interactions is filled with '1' in every element, the result that it is maximally redundant scales to strings of any length and character set.

Ubiquitous binding has trivial redundancy and no degeneracy, which makes the comparison in figure 4 appear unnecessary. However, this is a simple example of a general technique that, for a given alphabet, can be used to compare

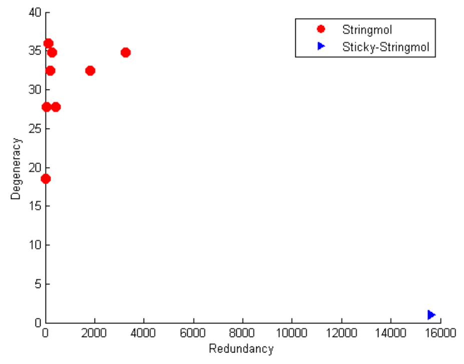


Figure 4: Comparison of the properties of the binding systems of Stringmol and sticky-Stringmol. The comparison shows the unscaled redundancy and degeneracy sets as the two systems contain the same number of elements. Each of the circles is associated with a redundant set. All of the circles together represent the Stringmol binding system. Sticky-Stringmol has only one triangle, as all of its elements form a single redundant set. It can be seen that the Stringmol binding system has a spread of both degeneracy and redundancy, whereas sticky-Stringmol's ubiquitous binding has only trivial redundancy.

two or more binding systems of any level of degeneracy and redundancy.

Degeneracy Affecting Simulation Artifacts

Stringmol is an AChem that encodes 'microprograms' as strings of characters. We give a brief overview of Stringmol here (for more details, see [11]). Each molecule is a string of characters that encodes a sequence of instructions, making use of pointer manipulations. A number of molecules are initialised in a reaction container. Pairs of molecules in the container are given an opportunity to react by the physics engine. In a biological system, although there may be thousands of different species of molecules, we note that in the majority of possible pairwise combinations, the number of molecules that each molecule interacts with is relatively small. As a result, care was taken in the design of Stringmol to check if two molecules could bind or not via a rich binding system. We made use of a tailored variant of the Smith-Waterman string-matching algorithm [11]. The Smith-Waterman algorithm is used in the study of Biology to compare the similarity of two sequences of DNA. In Stringmol, the Smith-Waterman based binding algorithm determines:

- with what probability two molecules bind;
- given that they bind, how the molecules are aligned;
- which molecule is the executing microprogram, and where its pointers are initialised.

Simulation Artifacts in Stringmol	Binding system property Degeneracy	No Degeneracy
Self replication	✓	✓
Parasites	✓	✓
Random walks	✓	✓
Sweeps	✓	×
Macro-mutations	✓	×
Hypercycles	✓	×
Parasite evasion*	✓	×

Table 1: Comparison of system level properties, used to evaluate the level of the complexity of an AChem. Degeneracy denotes the original Stringmol binding system, No Degeneracy denotes the ubiquitous binding system used in the sticky-Stringmol variant. * 'Parasite evasion' was not originally on the list of Stringmol's properties published in [10]; we introduce it and present evidence that it occurs in Stringmol, but not in sticky-Stringmol.

In 1000 trials of Stringmol, numerous phenomena were observed, including the emergence of hypercycles (two mutually dependent molecules), macro-mutations (non-point based mutations), sweeps (change of dominant replicase, other than by a random walk) and parasites [10].

The hypothesis is that: by changing the level of degeneracy in the binding system of an AChem, we will alter the simulation artifacts. We investigate this proposed link by comparing Stringmol and sticky-Stringmol (with thefull character set). The degeneracy and redundancy of these two binding systems (for a particular character set) is shown in figure 4. We repeated the experimental protocol of [10], running 500 trials of sticky-Stringmol to observe the diversity that arises from a mono-culture. Table 1 shows an comparison of the observed simulation artefacts. Sticky-Stringmol makes use of ubiquitous binding (no degeneracy), as opposed to the Smith-Waterman based binding of Stringmol (degeneracy).

The instruction set used in sticky-Stringmol is the same as the instruction used set in Stringmol. This might lead one to expect they should have computational artifacts of equal complexity; we find this is not the case, see table 1. These results show that a naive binding system, such as ubiquitous binding, can suppress complexity in an AChem. This identifies binding systems to potentially be an important aspect in all AChems.

These results indicate that the binding system has a strong effect of the overall complexity of the system.

Parasite Evasion in Stringmol Containers

Having presented the main results of the paper, we now present evidence of parasite evasion in Stringmol. For our purposes: a 'parasite' is a molecule that is replicated, but is unable to replicate other molecules in return. 'Parasite evasion' is when *the container survives* the introduction of a

```
$BLUBO^B>C$=?>$$BLUBO%}OYHOB
             |
$BLUBO^B>C$=?G$$BLUBO%}OYHOB
```

Figure 5: The functional regions of the replicase **R** (upper) and mutant **M** (lower). The location of the mutation is indicated by |. Both strings begin with the sequence ‘OBEQBXUTUDYGRHBBOREOLHHHRLUEUOBLROORE’ which is where the binding regions are located. The mutation has the effect of breaking the copy loop.

		Passive			
		R	**M**	**O**	**S**
Active	**R**	**R**	**M**	**-**	**S**
	M	**O**	**O**	**-**	**O**
	O	**-**	**-**	**-**	**-**
	S	**-**	**-**	**-**	**S**

Table 2: Interactions of: the replicase **R**; parasitic mutant **M**; product of the mutant **O**; the new strain of replicase that is immune to the parasite **S**. The body of the table shows the outcome of the reaction for each possible combination of active and passive molecules. Where the symbol ‘**-**’ appears instead of defined molecular species, it denotes no product formed.

parasite. We consider the container to have evaded the paeasite when no parasitic molecules (of that strain) remain in the container. Parasite evasion was not originally detected and explained in [10], which is why we now provide an overview of the phenomena. We outline the two mechanisms by which Stringmol containers can survive a parastite and demonstrate that these mechanisms are not available in sticky-Stringmol.

We re-examined previously published Stringmol results [10] and located examples of mid-run parasites that were non-fatal to the container. Here, we give details of one such parasitic molecule and how it interacts with the dominant replicase. We use this example as the basis of our parasite evasion scenario. Figure 5 shows the functional region of the original replicase **R** and the parasitic mutant **M**. The parasite does not implement the loop in the microprogram that allows characters on the bound molecule to be iteratively copied. When the parasite **M** is the executing microprogram, the product of the reaction is **O**, a string of length one: ‘O’.

We examined how the Stringmol container survives this parasite in the original trial [10]. We found a new strain of replicase arose via a mutation in the binding region of **R**. This new strain **S** is never the executing molecule in reactions with **R** or **M** and is thus immune to the parasite, see table 2. The new strain that averts the death of the container, **S**, would have taken over the container via a ‘sweep’, even in the absence of a parasitic mutant as it is always passive when reacting with **R**. In the original trial, both **R** and **M** die out relatively quickly and the new strain becomes dominant. Figure 7 shows results of this scenario depicting typical dynamics of cases where this parasite is fatal and where the system evolves a resistant strain of replicase. Hence the container can sometimes evade what is a potentially fatal parasite.

Trial condition	No. Escapes
Stringmol	32
Stringmol no mutation	21
Sticky-Stringmol	0
Sticky-Stringmol no mutation:	0

Table 3: Number of escapes from the parasite scenario out of a possible 100 for the four trial conditions.

We investigate the potency of the parasite in Stringmol in order to demonstrate that the parasite is potentially fatal. We also repeat this investigation for sticky-Stringmol and note it is unable to evade the parasite.

Our experimental setup initiates with two string types in the container: A replicase **R** and a parasitic mutant **M** of which there are 300 and 10 respectively at the start of each trial. The container is simulated until no molecules remain or until 0.5 million time steps. In each case we record if the container survived the parasite. We ran 100 trials for each of the four experimental conditions: Stringmol with and without mutation; sticky-Stringmol with and without mutation. The results are presented in table 3.

As we can see from table 3, it is possible for the Stringmol container to evade the parasite without mutation. In the absence of mutation it appears that the probability of binding between **R** and **M** being 0.66 is sufficient for the parasite to not to establish itself in the container in some cases, see figure 8 for typical dynamics. Stringmol (with mutation) is more successful at evading the parasite, see figure 7 for typical dynamics. There are two mechanisms by which the stringmol container can evade a (potentially fatal) parasite. One is by having a relatively low probability of binding, making it hard for new strains to establish themselves in the container. The second mechanism is the potential to mutate to a resistant strain of replicase. Looking at the results in table 3, it would appear that the dominant factor is the 0.66 chance of binding that the replicase, **R**, has with the parasite **M**.

Sticky-Stringmol appears unable to escape this parasite scenario with or without mutation. The ubiquitous binding at probability 1 causes the parasite to dominate the container every time. See figure 6 for dynamics that are typical of sticky-Stringmol both with and without mutation. Mutation in sticky-Stringmol offers no refuge from the parasite molecule. Because the binding is ubiquitous, a parasite is a parasite to all replicase molecules, rather than a limited subset of replicase molecules.

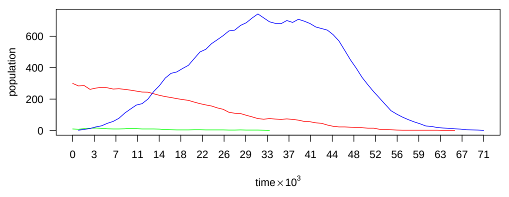


Figure 6: Sticky-Stringmol in the parasite evasion trial. The replicase, **R**, starts at the top at t=0. The mutant, **M**, starts at t=0 and maintains a low population. The product of the mutation, **O**, peaks in the middle of the run. This figure is representative behaviour of all 200 runs (both with and without mutation).

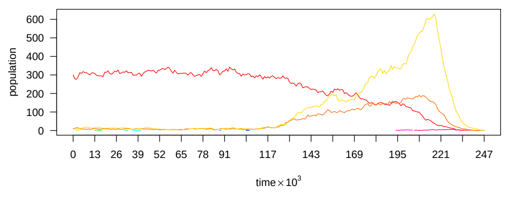
These results have a bearing on the main point of the paper, the importance on non-trivial binding, which is that it is not only which molecules bind to which that is important. The probability of binding also plays a role in determining the system level properties. Reducing the probability of binding from 1 to 0.66 does not simply cause the same outcome to happen more slowly. This is a refinement on our previous comments and highlights a limitation of our approach to characterizing degeneracy, which requires a boolean understanding of molecular interactions.

Discussion

Comparing the complexity of the artifacts in sticky-Stringmol with those of Stringmol, we note a loss of many of the more complex artifacts and no additional artifacts. These results demonstrate the importance of binding in AChems. They also indicate the potential for the complexity of a system to be stifled by a single naive component. This leads us to consider what other components of Stringmol (or any AChem) can have their levels of degeneracy measured and increased.

Investigations into the network properties of biological mutation networks, with an eye to how understanding their properties may lead to advances in ALife, are already underway [6]. That study makes use of network analysis techniques; our measure of degeneracy could be added to the array of such techniques. In cases where the sets A and B are the same, a binary interaction matrix specifies a network. Network analysis has a concept of ‘structural equivalence’ [15], which is the same as redundancy. The methods of measuring degeneracy and redundancy we introduce are also suitable for systems where $A \neq B$, which means they would also be applicable in other fields which do not have a natural mapping to a network, such as the binding of paratopes to epitopes in the immune system [2].

Much of the confusion regarding redundancy and degeneracy stems from the absence of a clear statement of context. A standalone statement of redundancy should take the form: ‘a_1 and a_2 are redundant given f, and in the context of B’. Statements of the truncated form: ‘a_1 and a_2 are redundant’ are ambiguous, relying on the author and reader to have an identical understanding of both f and B. Under some alternative criteria, f' and/or B', the elements a_1 and a_2 may well be redundant, degenerate or independent ($B_{a_1} \cap B_{a_2} = \emptyset$). If an author states both f and B explicitly, then the context of the redundancy is captured unambiguously. When presented with an ambiguous statement, the best one can do is assume the statement is true and attempt to determine in what context(s) this is the case, as this may give valuable insight.



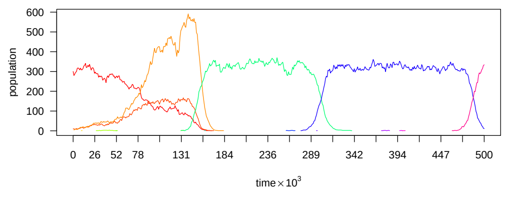


Figure 7: Stringmol (with mutation) in the parasite evasion trial. In both the upper and lower graphs the seed replicase, **R**, starts at the top at t=0. The upper graph shows a typical example of the dynamics when the parasite is lethal to the container. The 600 high spike towards the end of the run is the product of the parasite, **O**. The parasite, **M**, peaks at the same time as **O**, but to a height of only 200. The lower graph is an example where mutation gives rise to a new strain of replicase that is immune to the parasite and takes over the container. The 600 high spike is the product of the parasite, **O**. The parasite is fatal to the seed replicase **R**; but at the same time as the parasite and **O** are spiking, a new replicase molecule emerges. Typical Stringmol behavior can be seen for the remainder of the run, with two ‘sweeps’(where the dominant replicase is replaced by a mutation) occurring.

Conclusion

We have introduced definitions of degeneracy and redundancy that can be applied to individual examples, such as ‘are two elements degenerate or redundant’, in equations 3 and 4. We have also introduced definitions of degeneracy and redundancy that can be applied when talking about the levels of these properties within a set, in equations 6 and 7. Our measures of degeneracy and redundancy of sets have been defined in such a way that the concept of degeneracy is not conflated with redundancy.

We applied these measures to the binding system used in Stringmol [10][11], demonstrating that the binding sys-

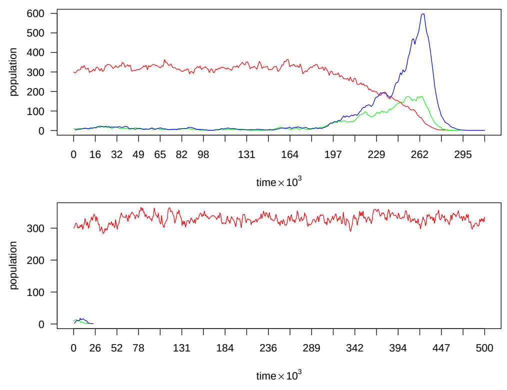


Figure 8: Stringmol without mutation in the parasite evasion trial. In both the upper and lower graphs the seed replicase, **R**, starts at the top at t=0. The upper graph shows a typical example of the dynamics when the parasite is lethal to the container. The 600 high spike towards the end of the run is the product of the parasite, **O**. The parasite, **M**, peaks at the same time as **O**, but to a height of only 200. The lower graph is an example of the parasitic mutant decaying out of the system, leaving the seed replicase, **R**, unaffected. The parasite **M**, and its product **O**, are present at t=0, but die out relatively quickly.

tem can produce both degeneracy and redundancy. We hypothesised the importance of a rich binding system to the complexity of the simulation artifacts Stringmol produces. We tested our hypothesis by constructing a deliberately poor Stringmol variant with ubiquitous binding, which we denote as ‘sticky-Stringmol’. Our measures show ubiquitous binding has no degeneracy. Our results demonstrated that a rich binding system facilitates many of the artifacts observed in Stringmol, including: hypercycles (two mutually dependent molecules), macro-mutations (non-point based mutations), sweeps (change of dominant replicase, other than by a random walk) [10], as well as parasite evasion. All of these complex phenomena were lost when we substituted Stringmol’s rich (degenerate) binding system for ubiquitous binding (no degeneracy).

We identified examples of parasite evasion in the previously published results of Stringmol [10], which we used as the basis of a parasite evasion experiment. We compared Stringmol to sticky-Stringmol in this parasite evasion trial, giving an overview of the mechanisms behind the loss of parasite evasion and demonstrating that the difference in binding system was the cause.

Though our hypothesis that ‘degeneracy in the components of an AChem facilitates the complexity of the system as a whole’ has not been proven in general terms, our results support this hypothesis and demonstrate the importance of binding systems in AChems.

Acknowledgements

This work is part of the Plazzmid project, funded by EPSRC grant EP/F031033/1

References

[1] W. Banzhaf, P. Dittrich, and H. Rauhe. Emergent computation by catalytic reactions. *Nanotechnology*, 7:307–314, 1996.

[2] Melvin Cohn. An in depth analysis of the concept of polyspecificity assumed to characterize tcr/bcr recognition. *Immunologic Research*, 40:128–147, 2008.

[3] James Decraene, George G. Mitchell, and Barry McMullin. Unexpected evolutionary dynamics in a string based artificial chemistry. In *ALife XI*, pages 158–165. MIT Press, 2008.

[4] P. Dittrich, J. Ziegler, and W. Banzhaf. Artificial chemistries–a review. *Artificial Life*, 7(3):225–275, 2001.

[5] Peter Dittrich and Pietro di Fenizio. Chemical Organisation Theory. *Bulletin of Mathematical Biology*, 69(4):1199–1231, 2007.

[6] A. P. Droop and S. J. Hickinbotham. Properties of biological mutation networks and their implications for alife. *Artificial Life*, 2011 (in press).

[7] Gerald M. Edelman and Joseph A. Gally. Degeneracy and complexity in biological systems. *PNAS*, 98(24):13763–13768, 2001.

[8] Adam Faulconbridge, Susan Stepney, Julian F. Miller, and Leo Caves. RBN-world: The hunt for a rich AChem. In *ALife XII*, pages 261–268. MIT Press, 2010.

[9] W. Fontana. Algorithmic Chemistry. In *ALife II*, pages 159–210. Addison Wesley, 1992.

[10] Simon Hickinbotham, Edward Clark, Susan Stepney, Tim Clarke, Adam Nellis, Mungo Pay, and Peter Young. Diversity from a monoculture: Effects of mutation-on-copy in a string-based artificial chemistry. In *ALife XII*, pages 24–31. MIT Press, 2010.

[11] Simon Hickinbotham, Edward Clark, Susan Stepney, Tim Clarke, Adam Nellis, Mungo Pay, and Peter Young. Specification of the stringmol chemical programming language version 0.1. Technical Report YCS-2010-457, Dept Computer Science, University of York, 2010.

[12] Raphael Plasson, Axel Brandenburg, Ludovic Jullien, and Hugues Bersini. Autocatalyses. In *ALife XII*, pages 4–11. MIT Press, 2010.

[13] G Tononi, O Sporns, and G M Edelman. A measure for brain complexity: relating functional segregation and integration in the nervous system. *PNAS*, 91(11):5033–5037, 1994.

[14] Giulio Tononi, Olaf Sporns, and Gerald M. Edelman. Measures of degeneracy and redundancy in biological networks. *PNAS*, 96(6):3257–3262, 1999.

[15] S. Wasserman and K. Faust. *Social Network Analysis: methods and applications*. Cambridge University Press, 1994.

[16] James Whitacre. Degeneracy: a link between evolvability, robustness and complexity in biological systems. *Theoretical Biology and Medical Modelling*, 7(1):6, 2010.