State aggregation and population dynamics in linear systems

Jonathan E. Rowe
School of Computer Science
University of Birmingham
Birmingham B15 2TT
Great Britain
J.E.Rowe@cs.bham.ac.uk

Michael D. Vose
Computer Science Department
University of Tennessee
Knoxville, TN 37996
USA
vose@cs.utk.edu

Alden H. Wright
Dept. of Computer Science
University of Montana
Missoula, Montana 59812
USA
wright@cs.umt.edu

Abstract

We consider complex systems that are comprised of many interacting elements, evolving under some dynamics. We are interested in characterising the ways in which these elements may be grouped into higher-level, macroscopic states in a way which is compatible with those dynamics. Such groupings may then be thought of as naturally emergent properties of the system. We formalise this idea, and, in the case that the dynamics are linear, prove necessary and sufficient conditions for this to happen. In cases where there is an underlying symmetry amongst the components of the system, group theory may be used to provide a strong sufficient condition. These observations are illustrated with three artificial life examples.

1   State aggregation and dynamical hierarchies

Many naturally occuring systems are comprised of a large collection of compenents, which interact with each other, and possibly with some background environment. These components often cluster into larger units: for example, DNA molecules in a cell nucleus, helium atoms in a balloon, people in a crowd. These units may themselves be components of some larger system. When performing simulations of such systems, rules are specified for the most basic components. It is hoped that, if the rules are correctly specified, then the behaviour of the higher-level units will emerge from these low-level interactions. An interesting question, which we begin to address in this paper, is to ask whether it is possible to deduce the rules of behaviour of the emergent structures, given the rules for

the basic components.[1] We would also like to know under what circumstances such emergent structures are non-trivial. For example, helium atoms in a balloon, and grains of sand in a pile [Gross and McMullen, 2001] seem intuitively to have at most a highly trivial hierarchical structure. Can this be formalised? We will show exactly why these examples are trivial, and characterise conditions for non-triviality.

We begin, in section 2, by defining, in a formal way, what is meant by the term "higher-level unit". We suggest that they should comprise aggregations of the basic components. That is, the set of underlying components will be partitioned into disjoint subsets, and each subset will then be considered as a unit in its own right. If we are given equations specifying the dynamics of the basic components, we want to be able to reconstruct the equations describing the dynamics of the subsets. If it is possible to do this, we will say that that particular way of partitioning the components into higher-level units is *compatible* with the dynamics of the components.

We will be modelling the dynamics of our system through a map that takes a *population* (that is, a collection of components in some configuration) at a given time step, and calculates the expected state at the next time step. In this paper, we will limit ourselves to the case where this map is *linear* (section 3). This case includes the description of systems forming a Markov process. The non-linear case will be investigated in a future paper. Aggregation on linear systems has previously been considered by [Shpak et al., 2003]. However, that paper assumes that the set of state variables describing a system is *decomposable* as well as *aggregable*. That is, they assume that the system can be decomposed into a partition of subsets, in which there is little or no interaction between the aggregated states (all interaction takes place within the subsets — not between them). We do not impose this restriction, and allow interactions between aggregated states to occur.

Before developing our theory, we present a simple example (section 4) which should clarify the issues. Section 5 explains how symmetries inherent in the system of underlying components can be described mathematically using elementary group theory. We then go on to show that the existence of such symmetries can be used to provide strong sufficient conditions for which a given state aggregation will be compatible with the underlying dynamics. This is the *group orbit theorem* which is proved in section 6.

We then present two examples which illustrate the application of our theorem. In section 7 we present a simplified model of molecules passing through a unidirectional membrane. In this example, the equations describing the dynamics can be dramatically simplified by using an appropriate hierarchy of state aggregation. Section 8 looks at the mutation of binary "DNA" strings (such as are typically used in genetic algorithms, for example). In this case, it turns out that we can characterize a range of higher-level units that are compatible with the action of mutating bits independently according to some mutation rate.

---

[1]We emphasise that we are considering whether or not the dynamics can be recovered *exactly*. Approximate methods for aggregating states in, say, a Markov process have been considered by other authors (see, for example, [Spears and Jong, 1997]).

Notation

We follow the convention that a logical expression in square brackets, $[expr]$, evaluates to 1 if the expression is true, and 0 otherwise.

2 Compatibility of equivalence relations

In this section we formalise our concept of aggregation, and define what it means for an aggregation to be compatible with the underlying dynamics of the system. To illustrate the definitions, we will use the following simple example, based on modelling voters' behaviour. We suppose there are three political parties: a centre party (C), a left wing party (L) and a right wing party (R). Between two elections, there is a chance that a voter will switch allegiance. A supporter of the centre party will stay loyal with probability 0.8, and switch to either of the other parties with probability 0.1 each. A left or right wing voter may vote for the centre party in the next election with probability 0.4, otherwise, they will stay loyal to their respective parties. The dynamics of the voters' behaviour from election to election is therefore described by the following Markov transition matrix:

	L	C	R
L	0.6	0.1	0
C	0.4	0.8	0.4
R	0	0.1	0.6

In general, we suppose that we have a set (or *population*) of N basic components in our system, and that each component may be in one of a number of states. The notion of a state is quite general. A state might be the position of a component in space; a disposition to react in a given way; an indication of how a component might be perceived by others, and so on. In our voting model, a state will be a voting choice (that is, one of $\{L, C, R\}$). We will assume that the set of possible states is finite. To preserve generality, we will simply number the states and refer to the set of all states as $\Omega = \{0, 1, 2, \ldots, n-1\}$. We will characterise a population of components by a *population vector*

$$p = (p_0, p_1, \ldots, p_{n-1})$$

in which p_k is the proportion of components in the population that have state $k \in \Omega$. Notice that this vector is independent of the population size N, so that

$$\sum_{k=0}^{n-1} p_k = 1$$

We define the *population state space* to be

$$\Lambda = \left\{ x \in \mathbb{R}^n : \sum_k x_k = 1 \text{ and } x_k \geq 0 \text{ for all } k \right\}$$

so that any population vector is an element of Λ. In the voting model, if an eighth of the voters choose L, a half choose C and the remaining three eighths choose R, then the population vector is $p = (0.125, 0.5, 0.375)$.

We suppose that the equations describing the dynamics of the basic components of the system are known. That is, we have a map $\mathcal{G} : \Lambda \to \Lambda$, from which we can reconstruct the dynamics of the system as follows:

1. If the system is **deterministic** then, given a population p at a given time step, $\mathcal{G}(p)$ is the population at the next time step.

2. If the system is **stochastic** then, given a population p at a given time step, $\mathcal{G}(p)$ is a *probability distribution* over the set of states Ω. This distribution is sampled N times to give the population at the next time step.

This model is the *random heuristic search* model developed by Vose [Vose, 1999]. In the voting model, the map $\mathcal{G}$ is given by the transition matrix shown above. In other words:

$$
\begin{aligned}
\mathcal{G}(x)_L &= 0.6x_L + 0.1x_C \\
\mathcal{G}(x)_C &= 0.4x_L + 0.8x_C + 0.4x_R \\
\mathcal{G}(x)_R &= 0.1x_C + 0.6x_R
\end{aligned}
$$

We are interested in defining a partition of the components. We begin by assuming there is an *equivalence relation*, $\equiv$, defined on Ω. That is, if $a, b \in \Omega$ are equivalent, $a \equiv b$, then we consider these two states to belong to the same higher-level unit. For example, we may consider that the two extremist parties L and R are sufficiently similar that they can be lumped together into an extremist set X. This is done by declaring that they are "equivalent". Such an equivalence relation can be extended to the vector space $\mathbb{R}^n$ as follows:

$$
x \equiv y \iff \sum_{j \in \Omega}[j \equiv i]x_j = \sum_{j \in \Omega}[j \equiv i]y_j \text{ for all representatives } i
$$

for any $x, y \in \mathbb{R}^n$. By "representative" we mean an element belonging to an equivalence class. Consider the components of a vector $x \in \mathbb{R}^n$ as assigning some weight to each element of Ω (that is, x_k is the weight assigned to k). Then this definition says that two vectors are equivalent if and only if they both assign the same total weight to each of the equivalence classes of Ω. In the voting model, we can say that two distributions of voters are equivalent if the same number of people vote for the extremist parties and the same number of people vote for the centre party in each distribution. That is $x \equiv y$ if $x_C = y_C$ and $x_L + x_R = y_L + y_R$.

We are now in a position to relate the partitioning of Ω (in terms of the equivalence classes determined by $\equiv$) to the dynamics given by $\mathcal{G}$.

Definition 1 *We say that a map $\mathcal{G} : \mathbb{R}^n \to \mathbb{R}^n$ is* compatible *with an equivalence relation $\equiv$ if*
$$
x \equiv y \implies \mathcal{G}(x) \equiv \mathcal{G}(y)
$$
for all $x, y \in \mathbb{R}^n$.

Two populations are equivalent if they assign the same weight to each equivalence class. If the map $\mathcal{G}$ is compatible, then we can follow the dynamics of just

the equivalence classes (the "higher-level units") without worrying about their microscopic details. That is, for a compatible map, we can follow the dynamics of the higher-level units as units in their own right. An interesting question therefore is, given a map $\mathcal{G}$ which determines the microscopic level of dynamics, what are the equivalence relations with which it is compatible? In other words, what are the naturally emergent higher-level units associated with the dynamics of the microscopic level?

In the voting model, we can directly verify that the aggregation of the two extremist parties is compatible with the dynamics. For if $x \equiv y$ are two equivalent distributions then $\mathcal{G}(x)_C = 0.4(x_L + x_R) + 0.8x_C = 0.4(y_L + y_R) + 0.8y_C = \mathcal{G}(y)_C$. Similarly, $\mathcal{G}(x)_L + \mathcal{G}(x)_R = 0.6x_L + 0.2x_C + 0.6x_R = 0.6(x_L + x_R) + 0.2x_C = 0.6(y_L + y_R) + 0.2y_C = \mathcal{G}(y)_L + \mathcal{G}(y)_R$.

3 Linearity and Markov processes

In this paper, we will restrict our attention to the case when the map $\mathcal{G}$ is linear. Notice that we are considering this map to be defined for the whole of $\mathbb{R}^n$ and not just to Λ. By linear is meant that for any vectors $x, y \in \mathbb{R}^n$ and for any real numbers $\lambda, \mu \in \mathbb{R}$

$$\mathcal{G}(\lambda x + \mu y) = \lambda \mathcal{G}(x) + \mu \mathcal{G}(y)$$

While this is an obvious limitation, it does include, for example, the case when the system is a Markov process and the map $\mathcal{G}$ is given by the transition matrix of the Markov chain. We will give three examples of simple artificial life simulations. The general non-linear case will be the subject of further work.

In the linear case, we can make some progress as follows.

Definition 2 *Given an equivalence relation $\equiv$ defined on Ω, let k be the number of equivalence classes. We define Ξ to be the $k \times n$ matrix with i, j entry*

$$\Xi_{i,j} = [i \equiv j]$$

(We take i to range over a set of equivalence class representatives).

The purpose of this matrix is to map a population, considered as a distribution over the underlying component states Ω, to a corresponding distribution over the higher-level units, given by equivalence classes.

For example, suppose $\Omega = \{0, 1, 2, 3, 4, 5\}$ and our equivalence relation induces a partition $\{0, 1, 2\}, \{3, 4\}, \{5\}$. We pick the elements $0, 3, 5$ to be representatives of each class. Then the matrix Ξ is given by the table:

$$\Xi = \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 1 & 1 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}$$

If we have a population vector $p = (0.1, 0.2, 0.1, 0.3, 0.15, 0.15)$ then $\Xi p = (0.4, 0.45, 0.15)$ which is the distribution over the equivalence classes.

Recall that the *kernel* of a linear operator is the subspace of vectors which are all mapped to 0 by the operator. So the kernel of Ξ is the subspace

$$\mathbf{Ker}\Xi = \{x \in \mathbb{R}^n : \Xi x = 0\}$$

(That this set forms a subspace of $\mathbb{R}^n$ can be easily checked).

We now have the following results:

Lemma 1 *Let $x, y \in \mathbb{R}^n$. Then $x \equiv y$ if and only if $x - y \in \mathbf{Ker}\Xi$.*

Proof

$$
\begin{aligned}
x \equiv y \quad &\Longleftrightarrow \quad \sum_{j \in \Omega} [j \equiv i]x_j = \sum_{j \in \Omega} [j \equiv i]y_j \text{ for all representatives } i \\
&\Longleftrightarrow \quad (\Xi x)_i = (\Xi y)_i \text{ for all } i \\
&\Longleftrightarrow \quad \Xi x = \Xi y \\
&\Longleftrightarrow \quad \Xi(x - y) = 0 \\
&\Longleftrightarrow \quad x - y \in \mathbf{Ker}\Xi
\end{aligned}
$$

$\square$

Theorem 2 *If $\mathcal{G} : \mathbb{R}^n \to \mathbb{R}^n$ is a linear map then it is compatible with $\equiv$ if and only if $\mathbf{Ker}\Xi$ is an invariant subspace of $\mathcal{G}$. That is, $\mathcal{G}(\mathbf{Ker}\Xi) \subseteq \mathbf{Ker}\Xi$.*

Proof

1. Suppose $\mathcal{G}$ is compatible, and let $v \in \mathbf{Ker}\Xi$. That is, $v \equiv 0$. Therefore, by compatibility, $\mathcal{G}(v) \equiv \mathcal{G}(0)$. But since $\mathcal{G}$ is linear, $\mathcal{G}(0) = 0$. Therefore $\mathcal{G}(v) \equiv 0$ so by the previous lemma, $\mathcal{G}(v) \in \mathbf{Ker}\Xi$.

2. Conversely, suppose that $\mathbf{Ker}\Xi$ is an invariant subspace of $\mathcal{G}$. Let $x \equiv y$, so that $x - y \in \mathbf{Ker}\Xi$. By hypothesis, $\mathcal{G}(x - y)$ must also be in $\mathbf{Ker}\Xi$. But $\mathcal{G}$ is linear, so $\mathcal{G}(x) - \mathcal{G}(y) \in \mathbf{Ker}\Xi$. Therefore $\mathcal{G}(x) \equiv \mathcal{G}(y)$ and so $\mathcal{G}$ is compatible with $\equiv$.

$\square$

It is helpful to notice that $\mathbf{Ker}\Xi$ is the set of all vectors that assign *zero* total weight to each equivalence class.

Returning once more to the voting example, in which the dynamics is given by a matrix (that is, a linear operator), the matrix Ξ takes the form

	L	C	R
C	0	1	0
X	1	0	1

$\mathbf{Ker}\Xi$ is the space of all vectors satisfying $x_C = 0$ and $x_L + x_R = 0$. For any vector $x \in \mathbf{Ker}\Xi$, we have

$$\mathcal{G}(x)_C = 0.4(x_L + x_R) + 0.8x_C = 0$$

and

$$\mathcal{G}(x)_L + \mathcal{G}(x)_R = 0.6x_L + 0.2x_C + 0.6x_R = 0.6(x_L + x_R) + 0.2x_C = 0$$

and so $x \in \mathbf{Ker}\Xi$ implies that $\mathcal{G}(x) \in \mathbf{Ker}\Xi$, which proves that the aggregation is compatible with the dynamics.

4 Example: migration

We take a very simple example to illustrate our definitions. Suppose we have a population of birds that spend most of the year on certain lakes, each bird living at a particular lake. There are some lakes in the north and some lakes in the south. Every winter, some of the birds from the north fly south, while birds from the south fly north. There is a small chance a bird will not migrate. The exact lake that a bird will end up at varies each trip. We might model this process stochastically as follows. For each lake in the north, we assign a probability distribution over the southern lakes. This gives the probability that a bird from that northern lake wil end up at any one of the southern likes. Similarly, for each southern lake, there is a probability distribution over the northern lakes. There is also a small probability of a bird remaining at the same lake.

For example, suppose there are five northern lakes N_1, N_2, N_3, N_4, N_5 and three southern lakes S_1, S_2, S_3 (see figure 1). We associate the states of our system with these lakes through a bijection with the set $\Omega = \{0, 1, 2, 3, 4, 5, 6, 7\}$ given by:

0	1	2	3	4	5	6	7
N_1	N_2	N_3	N_4	N_5	S_1	S_2	S_3

Our population of birds can be described by a population vector over these eight lakes.

Given a lake in the north, N_1 say, there is a probability distribution over S_1, S_2, S_3 describing the chance that a bird migrating from N_1 will arrive at each of these lakes. There is also a probability ϵ that the bird will remain at N_1.

We can therefore define a linear operator describing this process by:

$$
\begin{aligned}
M_{N_i, S_j} &= \quad \text{probability that bird will migrate from } S_j \text{ to } N_i \\
M_{S_j, N_i} &= \quad \text{probability that bird will migrate from } N_i \text{ to } S_j \\
M_{N_i, N_j} &= \quad \epsilon[i = j] \\
M_{S_i, S_j} &= \quad \epsilon[i = j]
\end{aligned}
$$

If $p \in \Lambda$ describes the population distribution over the lakes at a given time step, then Mp gives us a distribution which, when sampled an appropriate number of times (one for each bird) simulates what happens at the next time step.

It is clear that we can reduce this eight state system to a much simpler two state system, by aggregating together the northern and southern lake birds respectively. If a bird is in the north, there is a probability ϵ of remaining in the north. Otherwise it will fly south. Similarly for a bird currently in the south. We can define a linear operator for our two state system by

	N	S
N	ϵ	$1 - \epsilon$
S	$1 - \epsilon$	ϵ

The reason this aggregation is compatible with the dynamics of the individual birds is as follows. Suppose there are two different distributions x, y of birds which nevertheless assigned the same total number of birds to the north and

south. Then applying the map M to each of these would give us two new distributions Mx and My. However, Mx will assign the same amount of birds (probabilistically) to the north and south as My does. It doesn't matter exactly which of the northern and southern lakes has the birds — it is only the totals in each that matter.

The projection operator Ξ for this aggregation is

$$\Xi = \left[\begin{array}{cccccccc} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{array} \right]$$

So if we had a distribution of birds $p = (0.1, 0.2, 0.1, 0.05, 0.15, 0.2, 0.0, 0.2)$ then $\Xi p = (0.6, 0.4)$ gives the proportions of birds in the north and south respectively. The kernel of this operator, $\mathbf{Ker}\Xi$, is the set of all vectors giving zero weight to both northern and southern lakes. That is, the set of vectors x with the property $\Xi x = (0, 0)$.

The dynamics of the underlying system (given by the matrix M) is compatible with this aggregation. Our theorem tells us, then, that given any vector x in the kernel of Ξ then calculating Mx would give another vector, also in the kernel. We can check this by writing the matrix M in block form:

$$M = \left[\begin{array}{c|c} 0 & M_N \\ \hline M_S & 0 \end{array} \right] + \epsilon \mathbf{I}$$

where M_N is a 5×3 matrix giving the probability distributions for birds flying from north to south, and M_S is a 3×5 matrix giving the probability distributions for birds flying from south to north. $\mathbf{I}$ is the 8×8 identity matrix. Let $x \in \mathbf{Ker}\Xi$. We will write $x = (x_N, x_S)$ where x_N is a vector giving the distribution of birds in the north, and x_S gives the bird distribution for the south. The components of each of these sum to zero, since $\Xi x = (0, 0)$. Now

$$Mx = \left[\begin{array}{c|c} 0 & M_N \\ \hline M_S & 0 \end{array} \right] x + \epsilon \mathbf{I} x = \left[\begin{array}{c} M_S x_N \\ M_N x_S \end{array} \right] + \left[\begin{array}{c} \epsilon x_N \\ \epsilon x_S \end{array} \right]$$

Let us write $\Xi Mx = (u, v)$. Then

$$\begin{aligned} u &= \sum_i (M_S x_N)_i + \sum_i \epsilon(x_N)_i \\ &= \sum_i \sum_j (M_S)_{i,j}(x_N)_j \\ &= \sum_j (x_N)_j \sum_i (M_S)_{i,j} \\ &= (1 - \epsilon) \sum_j (x_N)_j \\ &= 0 \end{aligned}$$

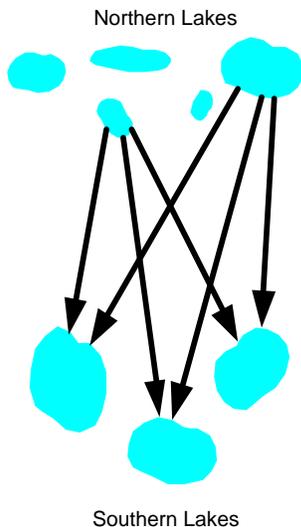Similarly, we can show $v = 0$. This confirms that $Mx \in \mathbf{Ker}\Xi$ as predicted by our theorem.

Figure 1: Birds migrate from northern to southern lakes and vice versa

5 Symmetry and group theory

Suppose that, in the migration example, there was just one probability distribution over the southern lakes. Then it wouldn't matter which of the northern lakes a bird sets off from — the probability of where it arrives is the same. There would then be a symmetry between each of the northern lakes, meaning that we could interchange them freely and the dynamics of the system would be unaffected. It is clear that all these lakes can be grouped together in that case. Obviously, this is a much stronger condition than we need to achieve a compatible partitioning. But it gives us a clue as to how we might prove some sufficient conditions for achieving compatibility.

Mathematically, symmetry is captured using group theory. One starts with an underlying set — in our case the set is Ω. One then considers permutations of this set (that is, bijections) which somehow leave the "shape" of the set unchanged. In geometry, these permutations might be rotations and reflections, for example. We establish a collection of permutation of Ω which we will call $L(\Omega)$ which has the following properties:

1. The identity map is in $L(\Omega)$.

2. If $a \in L(\Omega)$ then $a^{-1} \in L(\Omega)$.

3. If a and b are in $L(\Omega)$ then so is $a \circ b$.

The collection $L(\Omega)$ of permutations forms a *group*. The group is said to *act* on Ω.

The permutations represent the natural "folds" of the underlying set. They define a set of equivalence classes, called the *orbits* of the group action. For

each $i \in \Omega$ we define its orbit to be:

$$L(i) = \{j \in \Omega : j = a(i) \text{ for some } a \in L(\Omega)\}$$

Each orbit is an equivalence class of the equivalence relation

$$i \equiv j \iff j \in L(i)$$

In the following section, we will define a special group of permutations on Ω based on the properties of the map $\mathcal{G}$. This will lead to an equivalence relation, whose equivalence classes are guaranteed to be compatible with $\mathcal{G}$.

6 The group orbit theorem

Let a be a permutation of Ω. We associate with a the matrix σ_a with has i, j entry $[i = a(j)]$. It is simple to check that $\sigma_a^{-1} = \sigma_{a^{-1}}$ and that $\sigma_{a \circ b} = \sigma_a \circ \sigma_b$, for all permutations a, b.

Given $\mathcal{G} : \mathbb{R}^n \to \mathbb{R}^n$ is a linear map, define $H(\mathcal{G})$ to be the set of all permutations of Ω that *commute* with $\mathcal{G}$ in the sense that:

$$a \in H(\mathcal{G}) \iff \sigma_a \circ \mathcal{G} = \mathcal{G} \circ \sigma_a$$

Then we have the following:

Lemma 3 $H(\mathcal{G})$ *is a group action on Ω.*

Proof The identity is clearly in $H(\mathcal{G})$ since it commutes with any operator. If $a \in H(\mathcal{G})$ then, by definition, $\sigma_a \circ \mathcal{G} = \mathcal{G} \circ \sigma_a$, so by rearranging we see $\sigma_a^{-1} \circ \mathcal{G} = \mathcal{G} \circ \sigma_a^{-1}$ and therefore $\sigma_{a^{-1}} \circ \mathcal{G} = \mathcal{G} \circ \sigma_{a^{-1}}$. Finally, if $a, b \in H(\mathcal{G})$ then

$$
\begin{aligned}
\sigma_{a \circ b} \circ \mathcal{G} &= \sigma_a \circ \sigma_b \circ \mathcal{G} \\
&= \sigma_a \circ \mathcal{G} \circ \sigma_b \\
&= \mathcal{G} \circ \sigma_a \circ \sigma_b \\
&= \mathcal{G} \circ \sigma_{a \circ b}
\end{aligned}
$$

$\square$

Let G be a subgroup of $H(\mathcal{G})$, and define an equivalence relation on Ω by

$$i \equiv j \iff \exists a \in G, i = a(j)$$

That is, the equivalence classes are the orbits of the elements of G. We write the orbit of element i as $G(i)$. We extend this to an equivalence relation on $\mathbb{R}^n$

$$
\begin{aligned}
x \equiv y \quad &\Leftrightarrow \quad \forall t, \sum_i [i \equiv t] x_i = \sum_i [i \equiv t] y_i \\
&\Leftrightarrow \quad \forall t, \sum_i [i \in G(t)] x_i = \sum_i [i \in G(t)] y_i \\
&\Leftrightarrow \quad \forall t, \sum_i [i = a(t), \text{ some } a \in G] x_i = \sum_i [i = a(t), \text{ some } a \in G] y_i
\end{aligned}
$$

We want to count this sum over elements of G instead of over elements of Ω. However, there is the possibility of double counting. This occurs if there are two permutations $a, b \in G$ such that $a(t) = b(t) = i$. So, given any $i \in \Omega$, we need to know how many permutations map t to i. Write

$$G_{t,i} = \{a \in G : a(t) = i\}$$

Suppose $i \in G(t)$, and let $g \in G_{t,i}$. Consider the function

$$\phi : G_{t,t} \to G_{t,i}$$

given by

$$\phi(a) = g \circ a$$

This map ϕ is a bijection. (Since if $\phi(a) = \phi(b)$ then $g \circ a = g \circ b$ which gives $a = b$. And if $b \in G_{t,i}$, then $g^{-1} \circ b \in G_{t,t}$ and $\phi(g^{-1} \circ b) = b$). This means that the number of elements of $G_{t,i}$ is the same as the number of elements of $G_{t,t}$. That is, this number is independent of i. (This result is based on a theorem in [Biggs, 1971]).

Our condition for equivalence becomes:

$$x \equiv y \quad \Leftrightarrow \quad \forall t, \frac{\sum_{a \in G} x_{a(t)}}{|G_{t,t}|} = \frac{\sum_{a \in G} y_{a(t)}}{|G_{t,t}|}$$

$$\Leftrightarrow \quad \forall t, \sum_{a \in G} x_{a(t)} = \sum_{a \in G} y_{a(t)}$$

$$\Leftrightarrow \quad \forall t, \sum_{a \in G} (\sigma_{a^{-1}} x)_t = \sum_{a \in G} (\sigma_{a^{-1}} y)_t$$

$$\Leftrightarrow \quad \forall t, \sum_{a \in G} (\sigma_a x)_t = \sum_{a \in G} (\sigma_a y)_t$$

$$\Leftrightarrow \quad \forall t, (\sum_{a \in G} \sigma_a x)_t = (\sum_{a \in G} \sigma_a y)_t$$

$$\Leftrightarrow \quad \sum_{a \in G} \sigma_a x = \sum_{a \in G} \sigma_a y$$

Now suppose $x \equiv y$ and consider the effect of applying $\mathcal{G}$.

$$\sum_{a \in G} \sigma_a \mathcal{G}(x) \quad = \quad \sum_{a \in G} \mathcal{G}(\sigma_a x)$$

$$= \quad \mathcal{G}\left(\sum_{a \in G} \sigma_a x\right)$$

$$= \quad \mathcal{G}\left(\sum_{a \in G} \sigma_a y\right)$$

$$= \quad \sum_{a \in G} \mathcal{G}(\sigma_a y)$$

$$= \quad \sum_{a \in G} \sigma_a \mathcal{G}(y)$$

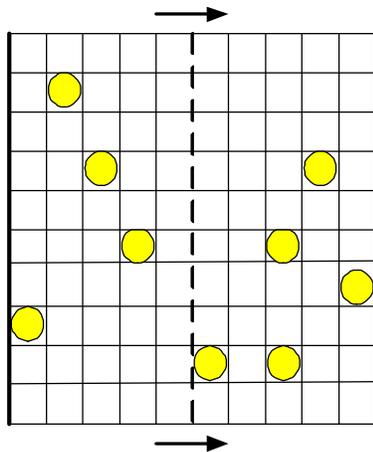and so $\mathcal{G}(x) \equiv \mathcal{G}(y)$. We have therefore proved

11

Figure 2: Molecules move around a tissue at random, but can only pass one way through the membrane

Theorem 4 *If $\mathcal{G} : \mathbb{R}^n \to \mathbb{R}^n$ is linear, and G is a subgroup of $H(\mathcal{G})$, then $\mathcal{G}$ is compatible with the equivalence relation given by the orbits of G acting on Ω.*

7   Example: unidirectional membranes

In this section we present a simplified model of molecular flow through a unidirectional membrane. This phenomenon occurs, for example, in Golgi bodies [Presley et al., 1998]. Molecules move around randomly within some confined space. The space is divided by a membrane which allows the molecules to pass through in one direction only. Obviously, we expect that eventually all molecules will end up on one side of the membrane. We wish to study the dynamics of this process, using our group orbit theorem to simplify the model.

The simulation set-up is shown in figure 2. There is a 10×10 grid on which the molecules wander at random. At each time step, a molecule picks a neighbouring square (north, south, east or west) to move to, uniformly at random. Molecules next to the left boundary cannot move further left: they only have a choice of three neighbours. Similarly, molecules next to the right boundary cannot move further right. It is assumed, though, that the top and bottom of the grid are connected (so the topology is cylindrical). There is a membrane running from top to bottom, half way across the grid. Molecules to the left of the membrane can move freely across it. However, molecules to the right are not allowed to pass back. Thus, molecules immediately to the right of the membrane also have a choice of only three neighbours.

Our initial model of this system requires 100 states. We label the squares on the grid from 0 to 99, counting from left to right, and top to bottom. The distribution of molecules in the system at any time step is given by a population vector $p = (p_0, p_1, \ldots, p_{99})$ where p_k is the proportion of molecules in square k. Each square k has a set of neighboring squares denoted $\nu(k)$ which are the legal moves allowed for a molecule at k. For example, $\nu(0) = \{1, 10, 90\}$. We

describe the evolution of the system by a linear map A given by:

$$A_{i,j} = \frac{[i \in \nu(j)]}{|\nu(j)|}$$

which gives the probability that a molecule moves from square j to square i. The matrix A is 100 by 100. We seek to use our theory to reduce this number of states by considering the symmetries that exist in the system.

For each $m = 0, 1, \ldots 9$ we define a bijection of $\Omega = \{0, 1, \ldots, 99\}$ by

$$a_m(k) = \begin{cases} k & \text{if } k \bmod 10 \neq m \\ k + 10 \bmod 100 & \text{otherwise} \end{cases}$$

The idea is that the number m indexes a column of the grid. The application of a_m to square k is to do nothing unless the square is in column m, in which case we shift down to the square beneath (wrapping round to the top if k is on the bottom row).

The collection $a_0, a_1, \ldots, a_9$ does not itself form a group. We also need to include the effects of repeatedly applying each map (a_m^r means apply a_m repeatedly r times) and of combining maps together ($a_m \circ a_{m'}$ means rotate both columns m and m'). This gives us the group *generated* by $a_0, a_1, \ldots, a_9$ (including the identity map). It is not hard to see that this group is commutative (that is, it does not matter what order each group element is applied). Therefore if the map A commutes with each of $a_0, a_1, \ldots, a_9$ then it commutes with the whole group. To ease notation we will denote by σ_m the matrix with i, j entry $[i = a_m(j)]$. Now

$$\begin{aligned} (\sigma_m A)_{i,j} &= \sum_k [i = a_m(k)] \frac{[k \in \nu(j)]}{|\nu(j)|} \\ &= \sum_k [a_m^{-1}(i) = k] \frac{[k \in \nu(j)]}{|\nu(j)|} \\ &= \frac{[a_m^{-1}(i) \in \nu(j)]}{|\nu(j)|} \end{aligned}$$

and

$$\begin{aligned} (A\sigma_m)_{i,j} &= \sum_k \frac{[i \in \nu(k)]}{|\nu(k)|} [k = a_m(j)] \\ &= \frac{[i \in \nu(a_m(j))]}{|\nu(a_m(j))|} \end{aligned}$$

It is clear from the rotational action of a_m that

$$|\nu(j)| = |\nu(a_m(j))|$$

and that

$$a_m(\nu(j)) = \nu(a_m(j))$$

and therefore $\sigma_m A = A \sigma_m$.

We therefore know that A commutes with the whole group and so will be compatible with its orbits. But what are the orbits of the group action? Applying

13

group members to a particular square either leaves it alone (if it is not in the appropriate column) or rotates it downwards. The orbits are therefore the columns of the grid. Our theorem tells us, therefore, that we can consider each column to be a high-level unit in its own right, and that we can write down the dynamics of the system in terms of these units. Accordingly, we number the columns giving a new set $\Omega = \{0, 1, \ldots, 9\}$. It is easy to calculate that the dynamics of the system at this new level of description is given by the linear operator

$$
C_{i,j} = \begin{cases}
[i = 0]2/3 + [i = 1]1/3 & \text{if } j = 0 \\
[i = 5]2/3 + [i = 6]1/3 & \text{if } j = 5 \\
[i = 8]1/3 + [i = 9]2/3 & \text{if } j = 9 \\
[i = j - 1]1/3 + [i = j]1/3 + [i = j + 1]1/3 & \text{else}
\end{cases}
$$

We have successfully reduced the number of states of the system from 100 to 10, by aggregating states into higher-level units. However, we can go further still, by recalling that once a molecule is passed the membrane, it cannot go back. We can therefore aggregate together all of the columns to the right of the membrane, and remain compatible with the dynamics. Our group orbit theorem does not apply here, however, as the probability of moving from column 4 to column 5 (that is, $1/3$) is not identical to the probability of moving to column 6, say (which is zero). Remember that the group orbit theorem gives a *sufficient* condition for compatibility, which is rather strong — it require all the incoming and outgoing probabilities to be identical. Nevertheless it is clear that we can successfully aggregate in this way, reducing the number of states to six. We number these $0, 1, 2, 3, 4, 5$, where state 5 represents all the columns to the right of the membrane; the other states are the corresponding left columns as before. The dynamics under this further reduction of states is given by the matrix:

$$
R = \begin{bmatrix}
2/3 & 1/3 & 0 & 0 & 0 & 0 \\
1/3 & 1/3 & 1/3 & 0 & 0 & 0 \\
0 & 1/3 & 1/3 & 1/3 & 0 & 0 \\
0 & 0 & 1/3 & 1/3 & 1/3 & 0 \\
0 & 0 & 0 & 1/3 & 1/3 & 0 \\
0 & 0 & 0 & 0 & 1/3 & 1
\end{bmatrix}
$$

We can then use standard Markov chain theory to predict the expected time for a molecule to cross the membrane, depending on which column it is starting in:

initial state	0	1	2	3	4
time to cross	45	42	36	27	15

8 Example: mutation of binary DNA

Our final example relates to the mutation of "DNA" which we represent as a fixed-length binary string (for example, this is common in genetic algorithms). Each bit in a string will be mutated with some fixed probability u. Our population comprises a pool of such strings, and we wish to track their evolution under this mutation operator.

The set of all binary strings of length ℓ can be conveniently identified with the set of integers $\Omega = \{0, 1, \ldots, 2^\ell - 1\}$ by interpreting the strings as being integers

written in base 2. The population at a given time step is therefore given by a vector $p = (p_0, p_1, \ldots, p_{2^\ell - 1})$. The dynamics of this system is given by a matrix U defined by

$$U_{i,j} = u^{h(i,j)}(1 - u)^{\ell - h(i,j)}$$

where $h(i, j)$ is the Hamming distance between strings i and j (that is, the number of bits at which they differ in value).

We wish to establish whether or not the set of all strings can be partitioned into higher-level units which are compatible with these dynamics. We are therefore concerned to identify the structure of the group $H(U)$.

Theorem 5 *Suppose Ω is identified with the set of binary strings of length ℓ and let U be the mutation matrix corresponding to bitwise mutation with rate μ. Then $H(U)$ is the group of the automorphisms of the hypercube.[2]*

Proof

The hypercube is the graph that has elements of Ω as vertices, with edges between vertices that differ in only one bit. Let $a \in H(U)$. We can view a as a permutation of vertices of the hypercube. Let $i, j \in \Omega$ be vertices that share an edge, so that $i \oplus j$ contains a single bit. We want to show that $a(i)$ and $a(j)$ also share an edge.

The fact that U commutes with a means that the probability of picking mask $i \oplus j$ is the same as picking $a(i) \oplus a(j)$. But when mutation is by a rate, then this can only depend on the number of ones in each mask. Thus if i and j differ in one bit, so do $a(i)$ and $a(j)$. This shows that $a \in H(A)$ is an automorphism of the hypercube.

Conversely, suppose π is an automorphism. Then the number of bits in $i \oplus j$ is the same as the number of bits in $\pi(i) \oplus \pi(j)$, for any $i, j \in \Omega$. Therefore the probability of picking $i \oplus j$ as a mask is the same as picking $a(i) \oplus a(j)$, and so U must commute with π.

$\square$

Now the automorphisms of the hypercube are generated by

1. masks (under exclusive-or)

2. permutations of bit positions

For example, suppose $\ell = 5$, and consider the set of strings

$$\{00000, 01000, 10000, 11000\}$$

. If we let each of these strings act on Ω as a mask under bitwise exclusive-or, then we obtain a subgroup of the automorphisms. This subgroup generates orbits whose members agree on the final three bits. Such a collection is referred to as a *schemata family* in the genetic algorithm literature [Rowe et al., 2002]. Another subgroup is the set of permutations of bit positions that shuffle the

---

[2] An *automorphism* of a graph is a permutation of the vertex set such that edges are mapped to edges. See [Biggs, 1971].

order of the first four bits. The orbits generated by this subgroup comprise sets whose strings contain the same number of 1s and 0s in the first four positions. Such collections are sometimes referred to as *unitation* classes. The fact that mutation is compatible with unitation classes can be exploited to investigate the evolution of asexual populations [Rowe, 1999, Vose and Rowe, 2000]. Other equivalence relations that are compatible with mutation include combinations of schemata and unitation classes. For example, we could collect together strings that agree with each other on bits 2 and 4, and have the same number of ones in bits 1, 3, 5. One such set is $\{01011, 01110, 11010\}$.

It should be noted that aggregation by schemata is also compatible with the dynamics of a variety of standard crossover operators, although here the dynamics are non-linear [Vose, 1999, Stephens and Waelbroeck, 1999]. Unitation classes are not compatible with crossover, however.

9 Conclusions and further work

We have presented a formalisation of what might be meant by the term "dynamical hierarchy". We view this as the situation in which the microscopic elements of a system can be clustered into higher-level units in such a way that the macroscopic dynamics is compatible with the rules of microscopic behaviour. Under these conditions, the higher-level units might be said to be naturally emergent properties of the system. Of course this will not work for arbitrary methods of clustering, and we have begun to investigate conditions under which this can be achieved. In the case where the underlying dynamics can be decribed as a linear function of the population state space, we have provided a necessary and sufficient condition for compatibility (namely, that the kernel of the corresponding projection is an invariant subspace of the dynamics). We have also proved a strong sufficient condition which exploits symmetries within the set of underlying components, using the language of group theory. In the general case, the microscopic dynamics may be described by a non-linear map $\mathcal{G}$. In this case, more work needs to be done to provide a characterisation of the compatible equivalence classes. The development of this theory will be the subject of a future paper.

If we return to the examples of the helium balloon and the sand-pile, we can now see exactly why their structures are trivial. In the case of the helium balloon, there are only two possible ways of aggregating the atoms which are compatible with the dynamics. Either each individual atom is in an equivalence class of its own, or we take the whole balloon to be a single aggregation. Of course, these two aggregations are always possible, in any system. The helium balloon has only trivial structure, because only the trivial aggregations are possible. The situation with the pile of sand is different, however. As long as the sand does not move, the *dynamics* of the system are trivial (that is, $\mathcal{G}$ is simply the identity map). This means that *any* aggregation of states will be compatible with the dynamics. We can partition the grains of sand into any subsets we like, and the resulting states will be compatible with the dynamics. This is the opposite extreme to the helium balloon example, but still renders the example trivial. If all ways of aggregating are possible, then there is no reason to prefer one way of doing it to another. We now see that the two examples are trivial, but for

completely different reasons. This observation enables us to characterise non-trivial dynamical hierarchies as being a subset system (ordered by inclusion) of compatible partitions, that is between these two extremes.

Acknowledgements

References

[Biggs, 1971] Biggs, N. L. (1971). *Finite groups of automorphisms*. Cambridge University Press, Cambridge, UK.

[Gross and McMullen, 2001] Gross, D. and McMullen, B. (2001). Is it the right ansatz? *Artificial Life*, 7(4):355–365.

[Presley et al., 1998] Presley, J. F., Smith, C., Hirschberg, K., Miller, C., and Cole, N. B. (1998). Golgi membrane dynamics. *Molecular Biology of the Cell*, 9(7):1617–1626.

[Rowe, 1999] Rowe, J. E. (1999). Population fixed-points for functions of unitation. In Banzhaf, W. and Reeves, C., editors, *Foundations of genetic algorithms (FOGA-5)*, San Mateo, CA. Morgan Kaufmann.

[Rowe et al., 2002] Rowe, J. E., Vose, M. D., and Wright, A. H. (2002). Group properties of crossover and mutation. *Evolutionary Computation*, 10(2):151–184.

[Shpak et al., 2003] Shpak, M., Stadler, P. F., Wagner, G. P., and Hermisson, J. (2003). Aggregation of variables and system decomposition: applications to fitness landscape analysis. Technical Report 03-04-025, Santa Fe Institute.

[Spears and Jong, 1997] Spears, W. M. and Jong, K. D. (1997). Analysing GAs using Markov models with semantically ordered and lumped states. In Belew, R. K. and Vose, M. D., editors, *Foundations of Genetic Algorithms*, volume 4, pages 85–100. Morgan Kaufmann.

[Stephens and Waelbroeck, 1999] Stephens, C. R. and Waelbroeck, H. (1999). Schemata evolution and building blocks. *Evolutionary Computation*, 7(2):109–124.

[Vose, 1999] Vose, M. D. (1999). *The Simple Genetic Algorithm: Foundations and Theory*. MIT Press, Cambridge, MA.

[Vose and Rowe, 2000] Vose, M. D. and Rowe, J. E. (2000). Random heuristic search: applications to GAs and functions of unitation. *Computer Methods in Applied Mechanics and Engineering.*, 186(2–4):195–220.