Theoretical Analysis of Generalised Recombination

Riccardo Poli
Department of Computer Science
University of Essex
UK
rpoli@essex.ac.uk

Christopher R. Stephens
Department of Computer Science
University of Essex
UK
csteph@essex.ac.uk

Department of Computer Science
University of Essex
Technical Report CSM-426
ISSN: 1744-8050
April 2005

Abstract

In this paper we propose, model theoretically and study a general notion of recombination for fixed-length strings where homologous crossover, inversion, gene duplication, gene deletion, diploidy and more are just special cases. The analysis of the model reveals similarities and differences between genetic systems based on these operations. It also reveals that the notion of schema emerges naturally from the model's equations even for the strangest of recombination operations. The study provides a variety of fixed points for the case where recombination is used alone, which generalise Geiringer's theorem.

1 Introduction

An important objective in evolutionary computation (EC) is to exactly model classes of evolutionary algorithms (EAs) and, further, to be able to draw inferences from these models that enhance theoretical understanding and, hopefully, aid "practitioners" in finding more competent EAs. Early models for GAs, proposed by Holland, Goldberg, Whitley and others in the seventies and eighties were either approximate or not easily scalable [4, 3, 21, 20]. Exact probabilistic models have been developed, such as the dynamical systems model of Vose and collaborators [19, 12]. More recently, an alternative exact approach, based on a coarse graining of the dynamics and directly involving schemata, has been introduced, leading to a spate of both new theoretical results [17, 15, 16, 7, 9, 10] and practical recipes for implementation [6, 8].

These models are important in that they allow for the mathematical investigation of the intrinsic dynamics of genetic systems, thereby nicely complementing, corroborating and, occasionally, disproving the findings of empirical studies. However, the vast majority of theoretical work in EAs, at least for classical fixed-length binary and real-valued representations, has been centered on the "canonical" genetic algorithm (GA) with selection, mutation and "homologous" recombination (where a locus in the offspring can by filled only by using alleles coming from the same locus in one of the parents). In nature, though, there are many more ways of combining parental genetic material into an offspring than just homologous crossover, many of which have been used in EAs. Gene duplication, for example, has been studied in biology [1] as well as in the context of GAs [13] and GP [5], while inversion was one of the operators used by Holland [4] in the original formulation of the GA.

In this paper we introduce an exact probabilistic model for fixed length strings, that extends current models by implementing a more general notion of recombination, that can account for *any*

distribution of the parental genes to the offspring, including as special cases, among others – fixed-length versions of gene duplication and deletion, as well as inversion and homologous crossover. We show that, as in the case of homologous crossover, a coarse graining naturally appears, revealing that the notion of schemata as building blocks emerges from the model's equations, even for the strangest of recombination operations. The analysis of the model reveals interesting similarities and differences between the various genetic operators present.

2 Generalised Recombination

Crossover masks are normally used to indicate from which parent to take an allele for each available locus. They are sufficient to model a crossover operator when only alleles at the same locus can be exchanged, i.e. homologous crossover. However, if we want to cope with other ways of redistributing genetic material, such as inversion, gene duplication, gene deletion, and, more generally, unequal crossing over, we need to allow for the possibility that the allele in one particular locus of the offspring comes from a different locus of a parent.[1]

This new level of generality can be represented mathematically in several equivalent ways. One is to use arrays (crossover matrices) instead of bit strings to represent crossover events. Crossover matrices are a generalisation of the notion of crossover mask. A crossover matrix will have as many rows as the number of loci in the offspring, say ℓ, and twice as many columns. The first ℓ columns indicate which alleles are copied from the first parent, while columns $\ell+1$ through to 2ℓ indicate what is provided by the second parent. The elements of the matrix are either 0 or 1. A 1 in row r and column c means that locus r in the offspring is filled with the allele in locus c in the first parent if $c \leq \ell$, or locus $c-\ell$ of the second parent otherwise. Because an offspring would not be fully specified if some of its alleles were undefined or would be overly specified if we tried to place more than one allele in a locus, in each row of a crossover matrix there must be exactly one 1 (with all other elements in the row being 0). For this reason we can also represent a recombination matrix as a vector $v = (v_1 \cdots v_\ell)$ with elements from $\mathcal{N}_{2\ell} = \{1, \cdots, 2\ell\}$, where v_i represents the position of the 1 in the i-th row. We will denote either the matrix or vector representation a Generalized Crossover Mask (GCM). The total number of GCMs is $(2\ell)^\ell$, many more than the 2^ℓ masks for homologous recombination. The action of a GCM, v, is then fully determined when the probability $p_c(v)$ of choosing any particular crossover matrix, or its equivalent crossover vector, is given. This is a generalisation of the notion of recombination distribution – the Generalized Recombination Distribution (GRD).

Another useful representation is a hybrid between the notion of crossover mask and the recombination vector. To represent a possible recombination event we use a *recombination pair* $r \equiv (m, v)$ where $m = (m_1 \cdots m_\ell)$ is an ℓ-component bit vector (i.e., $m \in \{0,1\}^\ell$) and $v = (v_1, \cdots, v_\ell)$ is a vector of integers whose components are in $\{1, \cdots, \ell\}$ (i.e., $v \in \mathcal{N}_\ell^\ell$). The semantics of this representation is very simple. The elements in m specify which parent contributes the alleles to fill each locus in the offspring, while the elements of v tell us which particular alleles in a parent will be transferred to the offspring. So, $m_i = 1$ means locus i will be filled with an allele from parent 1, $m_i = 0$ means parent 2 will contribute the allele instead. If the corresponding entry $v_i = j$ then locus i will be filled with the allele currently in position j in a parent. In this notation, traditional (homologous) crossover events can be represented with pairs of the form $r = (m, (1, 2, \cdots, \ell))$ where, effectively, m can be seen as a traditional crossover mask.

As an example of how the different representations of a GCM work consider the following example using standard one-point crossover for $\ell = 3$. The associated crossover matrices are

$$\begin{pmatrix} 1\,0\,0 & 0\,0\,0 \\ 0\,0\,0 & 1\,0\,0 \\ 0\,0\,0 & 0\,0\,1 \end{pmatrix} \quad \begin{pmatrix} 1\,0\,0 & 0\,0\,0 \\ 0\,1\,0 & 0\,0\,0 \\ 0\,0\,0 & 0\,0\,1 \end{pmatrix}$$

each invoked with probability $\frac{1}{2}$. These are equivalent to the recombination vectors $v_1 = (1, 5, 6)$

[1] This more general way of redistributing genetic material can also be used to model diploidy.

and $v_2 = (1, 2, 6)$, or to the recombination pairs $r_1 = (100, (1, 2, 3))$ and $r_2 = (110, (1, 2, 3))$, or to the more traditional crossover masks 100 and 110.

To see the large variety of ways in which parental genetic material can be distributed among the offspring consider the case of $\ell = 2$, where the $(2 \times 2)^2 = 16$ recombination pairs are

(00,(1,1))	(00,(1,2))	(00,(2,1))	(00,(2,2))
(01,(1,1))	(01,(1,2))	(01,(2,1))	(01,(2,2))
(10,(1,1))	(10,(1,2))	(10,(2,1))	(10,(2,2))
(11,(1,1))	(11,(1,2))	(11,(2,1))	(11,(2,2))

If the associated GRD is such that each is invoked with probability $p_c(m, v) = \frac{1}{16}$, this would represent a recombination operator where each locus in the offspring is filled with a randomly chosen allele from the parents. Clearly this operator could not be represented with crossover masks. As a final example, the following GRD represents a single-parent inversion operator in the case of a three-locus system:

$$p_c(111, (2, 1, 3)) = p_c(111, (1, 3, 2)) = p_c(111, (3, 2, 1)) = \frac{1}{3}$$

2.1 Mixing graph and recombination cliques

An important concept when considering redistribution of genetic material as determined by the GRD is: in which direction can one have a flow of genes? As qualitatively different behaviours are exhibited by genetic systems with different GRDs, to understand which features are important, we model the effects of the GRD through a *mixing graph*. The nodes in the graph represent different loci. The arcs are *directed* and represent causal relationships between loci. Thus, we will connect locus i with an arrow from locus j if the frequency of alleles in locus i can be influenced by the allele frequency of locus j. Figure 1 shows an example of a mixing graph for a 7-locus representation.

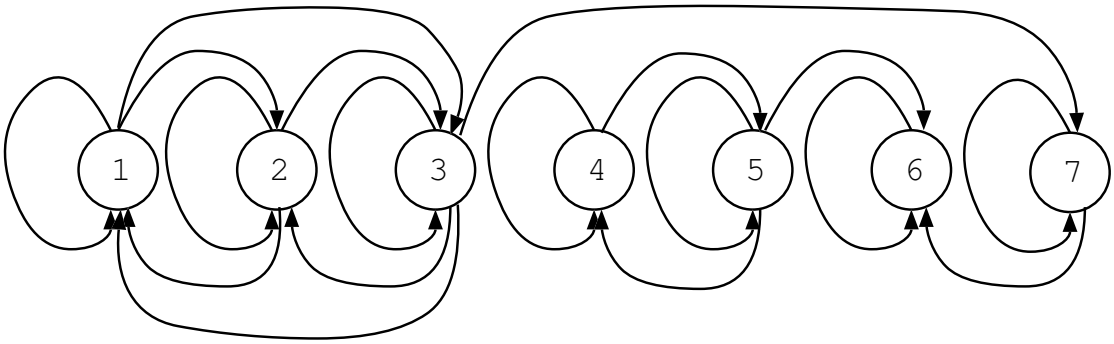

Figure 1: Example of mixing graph for $\ell = 7$.

The network of causal influences is completely determined by the GRD. The connection matrix $C = (c_{ij})$ for the mixing graph is given by

$$c_{ij} = \delta\big(p_c(*\cdots*, (\underbrace{*, \cdots, *}_{i-1}, j, \underbrace{*, \cdots, *}_{\ell-i})) > 0\big)$$

where $\delta(x) = 1$ if x is true, while $\delta(x) = 0$ otherwise. If there is a directed path between each pair of nodes in the mixing graph (the mixing graph is strongly connected), we say that the recombination is *fully mixing*.

Imagine a population of strings and focus attention on a particular allele a at a particular locus l of a particular string s. A fully-mixing generalised crossover allows for the migration of allele

a to different strings. So, generalised crossover promotes a process of "diffusion" of alleles from one locus to other loci. That is, unlike the case of homologous crossover, in general, generalised crossover does not keep the alleles in their original position, i.e. allele a might migrate to loci different from l. Because of this, in repeated applications of crossover, a copy of the allele can be placed back into the original string s (which may now have a different allele composition) but at a different locus, effectively creating a sort of gene duplication (indeed unequal crossing over seems to be the mechanism of gene duplication in nature [11]). Put another way, crossover is trying to spread each allele as thinly as possible over every locus available in the population. On the other hand, for homologous crossovers, the mixing matrix is diagonal and so each node in the graph is isolated (having only a self-connection).

Naturally, many qualitatively different intermediate situations are also possible. In all intermediate cases we can divide the mixing graph into two or more *recombination cliques*. These are characterised by the fact that all pairs of nodes in a clique are mutually accessible by traversing only nodes and arcs in the clique, while none of the nodes in a clique is mutually accessible from any other node outside the clique. In Figure 1, loci 1–3 form a recombination clique, nodes 4 and 5 form another, and nodes 6 and 7 form two single-node cliques. Formally, recombination cliques are the strong components of the recombination graph. So, each locus belongs to one and only one clique. Also, the cliques themselves form a directed acyclic graph (component graph) that we will call the *recombination clique graph*. This has one node for each recombination clique and an arc between two nodes if there is an edge between the corresponding cliques. Figure 2 shows an example of a recombination clique graph.

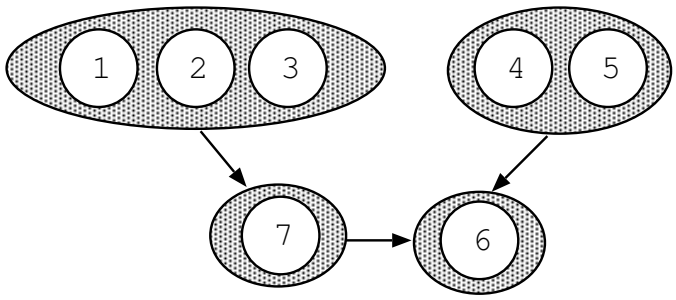

Figure 2: The recombination clique graph for the mixing graph in Figure 1.

3 Evolution equations

3.1 Evolution equations for strings

We will now derive and study exact equations for a generational evolutionary system based on selection and generalised recombination and using a fixed-length representation of size ℓ, where alleles take values from a generic alphabet Ω of any fixed cardinality. Under these assumptions the frequency of a string $h = h_1 \cdots h_\ell \in \Omega^\ell$ is given by

$$\begin{aligned} & E[\Phi(h, t+1)] \\ & \quad = \sum_{a \in \mathrm{P}(t)} p(a,t) \sum_{b \in \mathrm{P}(t)} p(b,t) \sum_{r \in \mathcal{R}_\ell^\ell} p_c(r) \gamma(a, b, r \to h) \end{aligned}$$

where $\mathrm{P}(t)$ is the population at generation t, $p(a,t)$ is the probability of picking a string of type a as a parent from such a population and $\mathcal{R}_\ell^\ell = \{0,1\}^\ell \times \mathcal{N}_\ell^\ell$ is the set of all possible crossover pairs. $p_c(r)$ is the GRD and $\gamma(a, b, r \to h)$ is the conditional probability that the offspring h is formed given the parents a and b and a GCM r. It returns value 1 if h is created from a and b using the GCM r and otherwise. Note that we can extend the string summations to cover the entire search

space Ω^ℓ rather than just the population $\mathrm{P}(t)$. We are allowed to do so on the assumption that the selection probability $p(x)$ of a string a in Ω^ℓ but not in $\mathrm{P}(t)$ is zero. Note, also, that the model is written in terms of the underlying microscopic degrees of freedom – the strings themselves. Note also that the equation is functionally identical to that for the case of standard mask-based crossover [14], the only difference being the different recombination distribution, and hence the different set of $\gamma(a, b, r \to h)$ that are non-zero. As in the standard crossover case, we have 2^ℓ coupled, first-order difference equations to solve. The chief problem, however, is the fact that on the right hand side we have, for binary strings, $2^\ell \times 2^\ell \times (2\ell)^\ell = (8\ell)^\ell$ contributing terms. For example, for two bits there are sixteen GCMs while the sums over the strings a and b run over the values 1 to $|\Omega^\ell|$. Thus, for an arbitrary GRD, even at the two bit level there are $16 \times 4 \times 4 = 256$ $\gamma(a, b, r \to h)$ to compute for a given string h. What is more, for a given h and r, there are potentially many different parental pairs a and b that can yield as offspring h.

In the case of homologous crossover these defects can be circumvented by coarse graining the dynamical equations and passing to a description in terms of Building Block Schemata rather than strings. One is naturally led to enquire as to whether similar benefits may be accrued in this more complex scenario.

The offspring $h = h_1 \cdots h_\ell$, produced by parents $a = a_1 \cdots a_\ell$ and $b = b_1 \cdots b_\ell$, with GRM $r = (m, v)$, can be represented very simply:

$$h_i = m_i a_{v_i} + (1 - m_i) b_{v_i}.$$

where a_{v_i} is the allele from the first parent picked out by the crossover pair r, and similarly for b_{v_i} from the second. Then

$$\gamma(a, b, r \to h) = \prod_{i \in I_r} \delta(h_i = a_{v_i}) \prod_{j \in \bar{I}_r} \delta(h_j = b_{v_j})$$

where $I_r = \{i : m_i = 1\}$ represents the genes picked out from the first parent by r that go to form part of the offspring h, and $\bar{I}_r = \{i : m_i = 0\}$ is the complementary set picked out from the second parent. As the full genetic composition of h has to come from the parents we have $I_r \cup \bar{I}_r = \{1, 2, \cdots, \ell\}$. By substituting this result into the evolution equation for h and reordering terms, we obtain

$$\begin{aligned} & E[\Phi(h, t+1)] \\ = & \sum_{r \in \mathcal{R}_\ell^\ell} p_c(r) \sum_{a \in \Omega^\ell} p(a, t) \prod_{i \in I_r} \delta(h_i = a_{v_i}) \\ & \sum_{b \in \Omega^\ell} p(b, t) \prod_{j \in \bar{I}_r} \delta(h_j = b_{v_j}). \end{aligned}$$

The effect of terms of the form $\prod_{i \in I_r} \delta(h_i = a_{v_i})$ in this equation is simply to limit the summations to subsets of Ω^ℓ. If we denote the elements of I_r with i_k (and the elements of $\bar{I}_r$ with j_k) and we use the standard computer science notation x^y to indicate pattern x repeated y times, these subsets are

$$\Gamma(h, I_r) = \bigcap_{k=1}^{|I_r|} (*^{v_{i_k} - 1} h_{i_k} *^{\ell - v_{i_k}})$$

and the corresponding $\Gamma(h, \bar{I}_r)$. Therefore

$$\begin{aligned} & \sum_{a \in \Omega^\ell} p(a, t) \prod_{i \in I_r} \delta(h_i = a_{v_i}) \\ = & \sum_{a \in \Gamma(h, I_r)} p(a, t) \\ = & \; p(\Gamma(h, I_r), t) \end{aligned}$$

Thus, we see that the action of the GCM r is to induce a coarse graining on the string sums. The benefit of this is immediately apparent, in that the string sum $\sum_{a\in\Omega^\ell}$ has disappeared. Thus, $p(\Gamma(h,I_r),t)$ denotes the probability for selecting the Building Block schema $\Gamma(h,I_r)$ which forms part of the offspring. In essence, this is identical to the case of homologous crossover. What is more complex here, in the presence of generalized recombination, is the form that the Building Block can take. For example, for $\ell=4$ and $r=(1101,(4,3,4,1))$, then $I_r=\{1,2,4\}$ and, so,

$$\begin{aligned}
&\Gamma(h_1h_2h_3h_4,\{1,2,4\}) = \\
&= (*^{v_{i_1}-1}h_{i_1}*^{4-v_{i_1}})\cap(*^{v_{i_2}-1}h_{i_2}*^{4-v_{i_2}})\cap \\
&\quad (*^{v_{i_3}-1}h_{i_3}*^{4-v_{i_3}}) \\
&= (*^3h_1)\cap(*^2h_2*)\cap(h_4*^3) \\
&= h_4*h_2h_1
\end{aligned}$$

hence the first Building Block for the string $h_1h_2h_3h_4$ for the above GRD is $h_4*h_2h_1$. The second Building Block is $\Gamma(h,\bar{I}_r)=***h_3$. Note that, unlike for the homologous case, in general $h\neq\Gamma(h,I_r)\cap\Gamma(h,I_r)=h_4*h_2h_1\cap***h_3$. This new notation based on schemata and the previous calculations lead us to the following

Theorem (Coarse-grained string evolution equation) *The expected frequency of a string h at the next generation in a generational GA with selection and generalized recombination is given by*

$$E[\Phi(h,t+1)] = \sum_{r\in\mathcal{R}_\ell^\ell} p_c(r)p(\Gamma(h,I_r),t)p(\Gamma(h,\bar{I}_r),t), \tag{1}$$

where $\Gamma(h,I_r)=\bigcap_{i\in I_r}H_{v_i}^{h_i}$, H_s^a is a shorthand notation for the generic order 1 schema $*^{s-1}a*^{\ell-s}$, and $\bar{I}_r=\{1,\cdots,\ell\}\setminus I_r$.

Thus, as in the case of homologous crossover, we see that evolution proceeds by building a string from its component Building Block schemata. Of course, to make further progress, one would then need to have the equations that govern these schemata.

3.2 Coarse-grained evolution equations

For homologous crossover, one of the most remarkable features of the coarse grained exact schema equations is their form invariance under a further coarse graining [17], i.e. that the functional form of the equations for a Building Block schema is identical to that of the equations for the strings themselves. This means that building blocks for a string are composed, in their turn, by other more coarse grained (lower order) building blocks, which in their turn etc., the whole hierarchy terminating at the 1-schemata. It is precisely the existence of this form invariance and the hierarchical nature of the relationship between the different building blocks that has led to so many new results using the coarse grained formulation. We are thus led to consider whether for generalized recombination the same features appear which can then be further exploited to gain a better theoretical understanding and derive new practical results. Thus, we begin by considering what happens when we coarse grain such that $h_1\cdots h_\ell\to\sum_{h_s}h_1\cdots h_s\cdots h_\ell=h_1\cdots*\cdots h_\ell$. Thus

$$
\begin{aligned}
&E[\Phi(h_1 \cdots h_{s-1} * h_{s+1} \cdots h_\ell, t+1)] \\
&= \sum_{h_s} E[\Phi(h_1 \cdots h_s \cdots h_\ell, t+1)] \\
&= \sum_{h_s} \sum_{r \in \mathcal{R}_\ell^\ell} p_c(r) p\left(\Gamma(h, I_r), t\right) p\left(\Gamma(h, \bar{I}_r), t\right) \\
&= \sum_{r \in \mathcal{R}_\ell^\ell} m_s p_c(r) p\left(\Gamma(h, \bar{I}_r), t\right) \sum_{h_s} p\left(\Gamma(h, I_r), t\right) \\
&+ \sum_{r \in \mathcal{R}_\ell^\ell} (1 - m_s) p_c(r) p\left(\Gamma(h, I_r), t\right) \sum_{h_s} p\left(\Gamma(h, \bar{I}_r), t\right)
\end{aligned}
$$

where $r \equiv (m, v)$ and m_s represents the s-th component of the bit string m.

If we use the notation $expr/y \leftarrow z$ to mean "replace every instance of y with z in expression $expr$", we can easily see that, if $s \in I_r$,

$$
p\left(\Gamma(h, I_r) \Big/ h_s \leftarrow *, t\right) = \sum_{h_s} p\left(\Gamma(h, I_r), t\right)
$$

If, instead, $s \notin I_r$, we have

$$
p\left(\Gamma(h, I_r) \Big/ h_s \leftarrow *, t\right) = p\left(\Gamma(h, I_r), t\right)
$$

The same applies to $\bar{I}_r$.

So, we can rewrite the above equation in the form

$$
\begin{aligned}
&E[\Phi(h_1 \cdots h_{s-1} * h_{s+1} \cdots h_\ell, t+1)] \\
&= E[\Phi(h \Big/ h_s \leftarrow *, t+1)] \\
&= \sum_{r \in \mathcal{R}_\ell^\ell} p_c(r) p\left(\Gamma(h, I_r) \Big/ h_s \leftarrow *, t\right) \\
&\quad p\left(\Gamma(h, \bar{I}_r) \Big/ h_s \leftarrow *, t\right)
\end{aligned} \tag{2}
$$

This derivation leads us to the following

Theorem (Schema evolution equation) *Equation 1 is applicable to both strings and schemata of any order.*

Proof With the previous calculations we have shown that Equation 1 is applicable to schemata with one "don't care" symbol. Since coarse graining over n variables can simply be obtained by coarse graining (over one variable) the evolution equations coarse-grained over $n-1$ variables, it follows that Equation 1 is applicable to schemata of with any number of don't care symbols. □

Interestingly, we can collect some terms in Equation 2. To see that let us assume, without loss of generality, that $m_s = 1$. In other words, $s \in I_r$. Let us assume, s is the n-th element of I_r, i.e., $i_n = s$. Therefore

$$
\begin{aligned}
&\Gamma(h, I_r) \Big/ h_s \leftarrow * \\
&= H_{v_{i_1}}^{h_{i_1}} \cap \cdots \cap H_{v_{i_n}}^{h_{i_n}} \cap H_{v_{i_{n+1}}}^{h_{i_{n+1}}} \cap \cdots \Big/ h_{i_n} \leftarrow * \\
&= H_{v_{i_1}}^{h_{i_1}} \cap \cdots \cap *^\ell \cap H_{v_{i_{n+1}}}^{h_{i_{n+1}}} \cap \cdots \\
&= \bigcap_{k=1, k \neq n}^{|I_r|} H_{v_{i_k}}^{h_{i_k}}
\end{aligned}
$$

since $*^\ell$ represents the whole search space Ω^ℓ.

A similar result holds for $s \in \bar{I}_r$. This means that neither m_s nor v_s appear explicitly in any of the terms in Equation 2, except $p_c(r)$. So, we can rewrite the equation as:

$$\begin{aligned} & E[\Phi(h_1 \cdots h_{s-1} * h_{s+1} \cdots h_\ell, t+1)] \\ = \quad & \sum p_c(m_1 \cdots m_{s-1} * m_{s+1} \cdots, (v_1 \cdots v_{s-1} * v_{s+1} \cdots)) \\ & p\left(\Gamma(h, I_r) \Big/ h_s \leftarrow *, t\right) p\left(\Gamma(h, \bar{I}_r) \Big/ h_s \leftarrow *, t\right) \end{aligned} \tag{3}$$

where the summation ranges over all GCMs $r = (m_1 \cdots m_{s-1} m_{s+1} \cdots m_\ell, (v_1 \cdots v_{s-1} v_{s+1} \cdots v_\ell)) \in \mathcal{R}_\ell^{\ell-1}$. Naturally this result generalises to any number of "don't care" symbols, leading to the following

Theorem *For a schema h with d don't care symbols at positions $l_1, \cdots, l_d$, the summation in Equation 1 can be turned into a summation over $(m', v') \in \mathcal{R}_\ell^{\ell-d}$ provided the recombination distribution p_c is replaced with the marginal distribution p'_c obtained by summing $p_c(m, v)$ over all m_{l_i} and v_{l_i} for $1 \le i \le d$.*

3.3 A more explicit notation

When, for a given recombination pair $r = (m, v) \in \mathcal{R}_\ell^\ell$, v is a permutation of the vector $(1, 2, \cdots, \ell)$, then $\Gamma(h, I_r) = \bigcap_{k=1}^{|I_v|} H_{v_{i_k}}^{h_{i_k}}$ is an ordinary schema. In order to be able to express exactly which schema this is we need to order the sets $I_r = \{i_1, i_2, \cdots, i_{|I_r|}\}$ and $\bar{I}_r = \{j_1, j_2, \cdots, j_{|\bar{I}_r|}\}$ on the basis of the corresponding entries in the vector v. That is, the elements i_k of I_r are ordered in such a way that $v_{i_k} \le v_{i_{k+1}}$ for any k, and the same is true for $\bar{I}_r$.

For example, if $\ell = 4$ and $r = (m, v) = (1101, (4, 3, 4, 1))$, then I_r is obtained as follows. As before, first we collect the indices of the elements of m that are 1 in a set (in this example, $\{1, 2, 4\}$). Then we sort the elements of this set based on the values of the corresponding elements in v. So, because $v_4 \le v_2 \le v_1$, $I_r = \{4, 2, 1\}$. Naturally, $\bar{I}_r = \{3\}$.

With this ordering, when v is a permutation, then $v_{i_k} < v_{i_{k+1}}$ for all k. Therefore

$$\Gamma(h, I_r) = \prod_{k=1}^{|I_r|} \left(*^{v_{i_k} - v_{i_{k-1}} - 1} h_{i_k}\right) *^{\ell - v_{i_{|I_r|}}}$$

where we used the convention that $v_{i_0} = 0$, that the $\prod$ operator means *concatenation* when applied to strings of symbols and that $*^0$ is the empty symbol (i.e. $*^0$ can be safely edited out from any sequence of characters).

We can interpret $\Gamma(h, I_r)$ as a schema also when $v_{i_k} = v_{i_{k-1}}$ for some k, as long as $h_{i_k} = h_{i_{k-1}}$. If this is not the case, then $\Gamma(h, I_r)$ is the empty set $\emptyset$ (naturally $p(\emptyset, t) = 0$). Therefore, in general we can write

$$\begin{aligned} & p\left(\Gamma(h, I_r), t\right) \\ = \quad & p\Big(\prod_{\substack{1 \le k \le |I_r| \\ i_k \ne i_{k-1}}} \left(*^{v_{i_k} - v_{i_{k-1}} - 1} h_{i_k}\right) *^{\ell - v_{i_{|I_r|}}}, t \Big) \\ & \times \prod_{\substack{1 \le k \le |I_r| \\ i_k = i_{k-1}}} \delta(h_{i_k} = h_{i_{k-1}}) \end{aligned} \tag{4}$$

4 Examples

4.1 $\ell = 2$

As an example, let us write the evolution equations for a generic string of length $\ell = 2$ from Equation 1 with the more explicit "δ notation" introduced in Section 3.3:

$$\begin{aligned} & E[\Phi(ab, t+1)] \\ = \;& p_{11}p(a*)\delta(a=b) + p_{12}p(ab) + p_{13}p(a*)p(b*) \\ + \;& p_{14}p(a*)p(*b) + p_{21}p(ba) + p_{22}p(*a)\delta(a=b) \\ + \;& p_{23}p(*a)p(b*) + p_{24}p(*a)p(*b) + p_{31}p(b*)p(a*) \\ + \;& p_{32}p(*b)p(a*) + p_{33}p(a*)\delta(a=b) + p_{34}p(ab) \\ + \;& p_{41}p(b*)p(*a) + p_{42}p(*b)p(*a) + p_{43}p(ba) \\ + \;& p_{44}p(*a)\delta(a=b) \end{aligned}$$

where for simplicity we omitted time from the selection probabilities and we used p_{ij} as a shorthand notation for the GRD $p_c(v)$, $v = (i, j) \in \mathcal{N}_4^2$ being a recombination vector (see Section 2).

If one replaces a and b with some values from Ω, all of the δ's turn either into 1's or 0's, and so it is possible to further simplify the equation. For example, if $a = b = 1$ and all GCMs have equal probability ($p_{ij} = 1/16$), we obtain

$$\begin{aligned} & E[\Phi(11, t+1)] \\ = \;& 0.125p(1*)^2 + 0.125p(1*) + 0.25p(1*)p(*1) \\ + \;& 0.125p(*1)^2 + 0.25p(11) + 0.125p(*1) \end{aligned}$$

Notice that in order to solve for the dynamics of the strings we need to have a solution for the building blocks `a*`, `*a`, `b*` and `*b`.

As an example, the evolution equation for the schema `a*` (a building block for `ab`) from Equation 3 is

$$\begin{aligned} E[\Phi(\texttt{a}*, t+1)] &= (p_{1*} + p_{3*})p(\texttt{a}*) \\ &+ (p_{2*} + p_{4*})p(*\texttt{a}) \end{aligned}$$

where $p_{x*} = \sum_y p_{xy}$.

A much deeper analysis of the $\ell = 2$ case is provided in [18], where a complete, exact solution, is derived, showing how the dynamical behaviour is radically different to that of homologous crossover. Even in such a simple case new qualitatively different behaviour is observed. For example, inversion is shown to potentially introduce oscillations in the dynamics, while gene duplication leads to an asymmetry between homogeneous and heterogeneous strings. Also, all non-homologous operators lead to allele "diffusion" along the chromosome.

4.2 $\ell = 3$

The general form of the evolution equations for $\ell = 3$ for the generic string `abc` is given if Figure 3. This includes 216 terms – a number that, although quite big, is only a tiny fraction of the number of terms one would get in the absence of coarse graining.

It is interesting to note that expected frequency of `abc` is a linear function of the selection probabilities of that string and all its permutations and a (generally) quadratic function of the selection probabilities of lower order schemata (building blocks). That is:

$$\begin{aligned} E[\Phi(\texttt{abc}, t+1)] &= p_{123}p(\texttt{abc}) + p_{132}p(\texttt{acb}) \\ &+ p_{213}p(\texttt{bac}) + p_{231}p(\texttt{cab}) \\ &+ p_{312}p(\texttt{bca}) + p_{321}p(\texttt{cba}) + b(t) \end{aligned}$$

Again, in order to solve for the string dynamics we need to have the dynamics of the building blocks that determine the driving term $b(t)$.

One of the building blocks, for example, is $\mathtt{ab*}$, the evolution equation of which is[2]

$$\begin{aligned}
&E[\Phi(\mathtt{ab}*, t+1)] \\
&= p_{11*}p(\mathtt{a}**)\delta(\mathtt{a}=\mathtt{b}) + p_{12*}p(\mathtt{ab}*) \\
&+ p_{13*}p(\mathtt{a}*\mathtt{b}) + p_{14*}p(\mathtt{a}**)p(\mathtt{b}**) \\
&+ p_{15*}p(\mathtt{a}**)p(*\mathtt{b}*) + p_{16*}p(\mathtt{a}**)p(**\mathtt{b}) \\
&+ p_{21*}p(\mathtt{ba}*) + p_{22*}p(*\mathtt{a}*)\delta(\mathtt{a}=\mathtt{b}) \\
&+ p_{23*}p(*\mathtt{ab}) + p_{24*}p(*\mathtt{a}*)p(\mathtt{b}**) \\
&+ p_{25*}p(*\mathtt{a}*)p(*\mathtt{b}*) + p_{26*}p(*\mathtt{a}*)p(**\mathtt{b}) \\
&+ p_{31*}p(\mathtt{b}*\mathtt{a}) + p_{32*}p(*\mathtt{ba}) \\
&+ p_{33*}p(**\mathtt{a})\delta(\mathtt{a}=\mathtt{b}) + p_{34*}p(**\mathtt{a})p(\mathtt{b}**) \\
&+ p_{35*}p(**\mathtt{a})p(*\mathtt{b}*) + p_{36*}p(**\mathtt{a})p(**\mathtt{b}) \\
&+ p_{41*}p(\mathtt{b}**)p(\mathtt{a}**) + p_{42*}p(*\mathtt{b}*)p(\mathtt{a}**) \\
&+ p_{43*}p(**\mathtt{b})p(\mathtt{a}**) + p_{44*}p(\mathtt{a}**)\delta(\mathtt{a}=\mathtt{b}) \\
&+ p_{45*}p(\mathtt{ab}*) + p_{46*}p(\mathtt{a}*\mathtt{b}) \\
&+ p_{51*}p(\mathtt{b}**)p(*\mathtt{a}*) + p_{52*}p(*\mathtt{b}*)p(*\mathtt{a}*) \\
&+ p_{53*}p(**\mathtt{b})p(*\mathtt{a}*) + p_{54*}p(\mathtt{ba}*) \\
&+ p_{55*}p(*\mathtt{a}*)\delta(\mathtt{a}=\mathtt{b}) + p_{56*}p(*\mathtt{ab}) \\
&+ p_{61*}p(\mathtt{b}**)p(**\mathtt{a}) + p_{62*}p(*\mathtt{b}*)p(**\mathtt{a}) \\
&+ p_{63*}p(**\mathtt{b})p(**\mathtt{a}) + p_{64*}p(\mathtt{b}*\mathtt{a}) \\
&+ p_{65*}p(*\mathtt{ba}) + p_{66*}p(**\mathtt{a})\delta(\mathtt{a}=\mathtt{b})
\end{aligned}$$

where we collected terms involving the same schemata and where $p_{xy*} = \sum_z p_{xyz}$.

Notice that the building block $\mathtt{ab*}$, in its turn, will depend on the dynamics of its own building blocks, such as $\mathtt{a**}$, the equation for which is

$$\begin{aligned}
E[\Phi(\mathtt{a}**, t+1)] &= (p_{1**} + p_{4**})p(\mathtt{a}**) \\
&+ (p_{2**} + p_{5**})p(*\mathtt{a}*) \\
&+ (p_{3**} + p_{6**})p(**\mathtt{a})
\end{aligned}$$

4.3 General case

These examples show that all schema/string evolution equations have the same structure

$$x(t+1) = Ax(t) + b(t)$$

with a linear part which depends on the selection probabilities of schemata of the same order as the schema on the left-hand side of the equation, and a non-linear forcing term $b(t)$ which depends on lower-order schemata. The only exception to this is order one objects, in which case $b(t) \equiv 0$. These objects, therefore, evolve independently but contribute to all higher-order schemata. So, *order one schemata act as pacemakers for a genetic system evolving under generalised recombination.* For these reasons we will analyse the evolution equations for such a case in more detail in the next section.

[2]Care must be taken when applying equations including δ's to schemata. This is can always be done but not necessarily by directly replacing defining symbols with "don't care" symbols. This works for Equation 1, but does not necessarily work when using the formalism in Equation 4. To apply equations expressed in the "δ formalism" to a schema one must either sum the string evolution equations and then collect terms, or apply Equation 1 to the schema and then transform the selection probabilities $p(\Gamma(\cdot,\cdot), t)$ into the δ notation.

$$
\begin{aligned}
& E[\Phi(\mathtt{abc}, t+1)] \\
= \;& p_{111}p(\mathtt{a}**)\delta(\mathtt{a}=\mathtt{b})\delta(\mathtt{a}=\mathtt{c}) + p_{112}p(\mathtt{ac}*)\delta(\mathtt{a}=\mathtt{b}) + p_{113}p(\mathtt{a}*\mathtt{c})\delta(\mathtt{a}=\mathtt{b}) + p_{114}p(\mathtt{a}**)p(\mathtt{c}**)\delta(\mathtt{a}=\mathtt{b}) \\
+ \;& p_{115}p(\mathtt{a}**)p(*\mathtt{c}*)\delta(\mathtt{a}=\mathtt{b}) + p_{116}p(\mathtt{a}**)p(**\mathtt{c})\delta(\mathtt{a}=\mathtt{b}) + p_{121}p(\mathtt{ab}*)\delta(\mathtt{a}=\mathtt{c}) + p_{122}p(\mathtt{ab}*)\delta(\mathtt{b}=\mathtt{c}) \\
+ \;& p_{123}p(\mathtt{abc}) + p_{124}p(\mathtt{ab}*)p(\mathtt{c}**) + p_{125}p(\mathtt{ab}*)p(*\mathtt{c}*) + p_{126}p(\mathtt{ab}*)p(**\mathtt{c}) \\
+ \;& p_{131}p(\mathtt{a}*\mathtt{b})\delta(\mathtt{a}=\mathtt{c}) + p_{132}p(\mathtt{acb}) + p_{133}p(\mathtt{a}*\mathtt{b})\delta(\mathtt{b}=\mathtt{c}) \\
+ \;& p_{134}p(\mathtt{a}*\mathtt{b})p(\mathtt{c}**) + p_{135}p(\mathtt{a}*\mathtt{b})p(*\mathtt{c}*) + p_{136}p(\mathtt{a}*\mathtt{b})p(**\mathtt{c}) + p_{141}p(\mathtt{a}**)p(\mathtt{b}**)\delta(\mathtt{a}=\mathtt{c}) \\
+ \;& p_{142}p(\mathtt{ac}*)p(\mathtt{b}**) + p_{143}p(\mathtt{a}*\mathtt{c})p(\mathtt{b}**) + p_{144}p(\mathtt{a}**)p(\mathtt{b}**)\delta(\mathtt{b}=\mathtt{c}) + p_{145}p(\mathtt{a}**)p(\mathtt{bc}*) \\
+ \;& p_{146}p(\mathtt{a}**)p(\mathtt{b}*\mathtt{c}) + p_{151}p(\mathtt{a}**)p(*\mathtt{b}*)\delta(\mathtt{a}=\mathtt{c}) + p_{152}p(\mathtt{ac}*)p(*\mathtt{b}*) + p_{153}p(\mathtt{a}*\mathtt{c})p(*\mathtt{b}*) \\
+ \;& p_{154}p(\mathtt{a}**)p(\mathtt{cb}*) + p_{155}p(\mathtt{a}**)p(*\mathtt{b}*)\delta(\mathtt{b}=\mathtt{c}) + p_{156}p(\mathtt{a}**)p(*\mathtt{bc}) + p_{161}p(\mathtt{a}**)p(**\mathtt{b})\delta(\mathtt{a}=\mathtt{c}) \\
+ \;& p_{162}p(\mathtt{ac}*)p(**\mathtt{b}) + p_{163}p(\mathtt{a}*\mathtt{c})p(**\mathtt{b}) + p_{164}p(\mathtt{a}**)p(\mathtt{c}*\mathtt{b}) + p_{165}p(\mathtt{a}**)p(*\mathtt{cb}) \\
+ \;& p_{166}p(\mathtt{a}**)p(**\mathtt{b})\delta(\mathtt{b}=\mathtt{c}) + p_{211}p(\mathtt{ba}*)\delta(\mathtt{b}=\mathtt{c}) + p_{212}p(\mathtt{ba}*)\delta(\mathtt{a}=\mathtt{c}) + p_{213}p(\mathtt{bac}) \\
+ \;& p_{214}p(\mathtt{ba}*)p(\mathtt{c}**) + p_{215}p(\mathtt{ba}*)p(*\mathtt{c}*) + p_{216}p(\mathtt{ba}*)p(**\mathtt{c}) + p_{221}p(\mathtt{ca}*)\delta(\mathtt{a}=\mathtt{b}) \\
+ \;& p_{222}p(*\mathtt{a}*)\delta(\mathtt{a}=\mathtt{b})\delta(\mathtt{a}=\mathtt{c}) + p_{223}p(*\mathtt{ac})\delta(\mathtt{a}=\mathtt{b}) + p_{224}p(*\mathtt{a}*)p(\mathtt{c}**)\delta(\mathtt{a}=\mathtt{b}) + p_{225}p(*\mathtt{a}*)p(*\mathtt{c}*)\delta(\mathtt{a}=\mathtt{b}) \\
+ \;& p_{226}p(*\mathtt{a}*)p(**\mathtt{c})\delta(\mathtt{a}=\mathtt{b}) + p_{231}p(\mathtt{cab}) + p_{232}p(*\mathtt{ab})\delta(\mathtt{a}=\mathtt{c}) + p_{233}p(*\mathtt{ab})\delta(\mathtt{b}=\mathtt{c}) \\
+ \;& p_{234}p(*\mathtt{ab})p(\mathtt{c}**) + p_{235}p(*\mathtt{ab})p(*\mathtt{c}*) + p_{236}p(*\mathtt{ab})p(**\mathtt{c}) + p_{241}p(\mathtt{ca}*)p(\mathtt{b}**) \\
+ \;& p_{242}p(*\mathtt{a}*)p(\mathtt{b}**)\delta(\mathtt{a}=\mathtt{c}) + p_{243}p(*\mathtt{ac})p(\mathtt{b}**) + p_{244}p(*\mathtt{a}*)p(\mathtt{b}**)\delta(\mathtt{b}=\mathtt{c}) + p_{245}p(*\mathtt{a}*)p(\mathtt{bc}*) \\
+ \;& p_{246}p(*\mathtt{a}*)p(\mathtt{b}*\mathtt{c}) + p_{251}p(\mathtt{ca}*)p(*\mathtt{b}*) + p_{252}p(*\mathtt{a}*)p(*\mathtt{b}*)\delta(\mathtt{a}=\mathtt{c}) + p_{253}p(*\mathtt{ac})p(*\mathtt{b}*) \\
+ \;& p_{254}p(*\mathtt{a}*)p(\mathtt{cb}*) + p_{255}p(*\mathtt{a}*)p(*\mathtt{b}*)\delta(\mathtt{b}=\mathtt{c}) + p_{256}p(*\mathtt{a}*)p(*\mathtt{bc}) + p_{261}p(\mathtt{ca}*)p(**\mathtt{b}) \\
+ \;& p_{262}p(*\mathtt{a}*)p(**\mathtt{b})\delta(\mathtt{a}=\mathtt{c}) + p_{263}p(*\mathtt{ac})p(**\mathtt{b}) + p_{264}p(*\mathtt{a}*)p(\mathtt{c}*\mathtt{b}) + p_{265}p(*\mathtt{a}*)p(*\mathtt{cb}) \\
+ \;& p_{266}p(*\mathtt{a}*)p(**\mathtt{b})\delta(\mathtt{b}=\mathtt{c}) + p_{311}p(\mathtt{b}*\mathtt{a})\delta(\mathtt{b}=\mathtt{c}) + p_{312}p(\mathtt{bca}) + p_{313}p(\mathtt{b}*\mathtt{a})\delta(\mathtt{a}=\mathtt{c}) \\
+ \;& p_{314}p(\mathtt{b}*\mathtt{a})p(\mathtt{c}**) + p_{315}p(\mathtt{b}*\mathtt{a})p(*\mathtt{c}*) + p_{316}p(\mathtt{b}*\mathtt{a})p(**\mathtt{c}) + p_{321}p(\mathtt{cba}) \\
+ \;& p_{322}p(*\mathtt{ba})\delta(\mathtt{b}=\mathtt{c}) + p_{323}p(*\mathtt{ba})\delta(\mathtt{a}=\mathtt{c}) + p_{324}p(*\mathtt{ba})p(\mathtt{c}**) + p_{325}p(*\mathtt{ba})p(*\mathtt{c}*) \\
+ \;& p_{326}p(*\mathtt{ba})p(**\mathtt{c}) + p_{331}p(\mathtt{c}*\mathtt{a})\delta(\mathtt{a}=\mathtt{b}) + p_{332}p(*\mathtt{ca})\delta(\mathtt{a}=\mathtt{b}) + p_{333}p(**\mathtt{a})\delta(\mathtt{a}=\mathtt{b})\delta(\mathtt{a}=\mathtt{c}) \\
+ \;& p_{334}p(**\mathtt{a})p(\mathtt{c}**)\delta(\mathtt{a}=\mathtt{b}) + p_{335}p(**\mathtt{a})p(*\mathtt{c}*)\delta(\mathtt{a}=\mathtt{b}) + p_{336}p(**\mathtt{a})p(**\mathtt{c})\delta(\mathtt{a}=\mathtt{b}) + p_{341}p(\mathtt{c}*\mathtt{a})p(\mathtt{b}**) \\
+ \;& p_{342}p(*\mathtt{ca})p(\mathtt{b}**) + p_{343}p(**\mathtt{a})p(\mathtt{b}**)\delta(\mathtt{a}=\mathtt{c}) + p_{344}p(**\mathtt{a})p(\mathtt{b}**)\delta(\mathtt{b}=\mathtt{c}) + p_{345}p(**\mathtt{a})p(\mathtt{bc}*) \\
+ \;& p_{346}p(**\mathtt{a})p(\mathtt{b}*\mathtt{c}) + p_{351}p(\mathtt{c}*\mathtt{a})p(*\mathtt{b}*) + p_{352}p(*\mathtt{ca})p(*\mathtt{b}*) + p_{353}p(**\mathtt{a})p(*\mathtt{b}*)\delta(\mathtt{a}=\mathtt{c}) \\
+ \;& p_{354}p(**\mathtt{a})p(\mathtt{cb}*) + p_{355}p(**\mathtt{a})p(*\mathtt{b}*)\delta(\mathtt{b}=\mathtt{c}) + p_{356}p(**\mathtt{a})p(*\mathtt{bc}) + p_{361}p(\mathtt{c}*\mathtt{a})p(**\mathtt{b}) \\
+ \;& p_{362}p(*\mathtt{ca})p(**\mathtt{b}) + p_{363}p(**\mathtt{a})p(**\mathtt{b})\delta(\mathtt{a}=\mathtt{c}) + p_{364}p(**\mathtt{a})p(\mathtt{c}*\mathtt{b}) + p_{365}p(**\mathtt{a})p(*\mathtt{cb}) \\
+ \;& p_{366}p(**\mathtt{a})p(**\mathtt{b})\delta(\mathtt{b}=\mathtt{c}) + p_{411}p(\mathtt{b}**)p(\mathtt{a}**)\delta(\mathtt{b}=\mathtt{c}) + p_{412}p(\mathtt{bc}*)p(\mathtt{a}**) + p_{413}p(\mathtt{b}*\mathtt{c})p(\mathtt{a}**) \\
+ \;& p_{414}p(\mathtt{b}**)p(\mathtt{a}**)\delta(\mathtt{a}=\mathtt{c}) + p_{415}p(\mathtt{b}**)p(\mathtt{ac}*) + p_{416}p(\mathtt{b}**)p(\mathtt{a}*\mathtt{c}) + p_{421}p(\mathtt{cb}*)p(\mathtt{a}**) \\
+ \;& p_{422}p(*\mathtt{b}*)p(\mathtt{a}**)\delta(\mathtt{b}=\mathtt{c}) + p_{423}p(*\mathtt{bc})p(\mathtt{a}**) + p_{424}p(*\mathtt{b}*)p(\mathtt{a}**)\delta(\mathtt{a}=\mathtt{c}) + p_{425}p(*\mathtt{b}*)p(\mathtt{ac}*) \\
+ \;& p_{426}p(*\mathtt{b}*)p(\mathtt{a}*\mathtt{c}) + p_{431}p(\mathtt{c}*\mathtt{b})p(\mathtt{a}**) + p_{432}p(*\mathtt{cb})p(\mathtt{a}**) + p_{433}p(**\mathtt{b})p(\mathtt{a}**)\delta(\mathtt{b}=\mathtt{c}) \\
+ \;& p_{434}p(**\mathtt{b})p(\mathtt{a}**)\delta(\mathtt{a}=\mathtt{c}) + p_{435}p(**\mathtt{b})p(\mathtt{ac}*) + p_{436}p(**\mathtt{b})p(\mathtt{a}*\mathtt{c}) + p_{441}p(\mathtt{c}**)p(\mathtt{a}**)\delta(\mathtt{a}=\mathtt{b}) \\
+ \;& p_{442}p(*\mathtt{c}*)p(\mathtt{a}**)\delta(\mathtt{a}=\mathtt{b}) + p_{443}p(**\mathtt{c})p(\mathtt{a}**)\delta(\mathtt{a}=\mathtt{b}) + p_{444}p(\mathtt{a}**)\delta(\mathtt{a}=\mathtt{b})\delta(\mathtt{a}=\mathtt{c}) + p_{445}p(\mathtt{ac}*)\delta(\mathtt{a}=\mathtt{b}) \\
+ \;& p_{446}p(\mathtt{a}*\mathtt{c})\delta(\mathtt{a}=\mathtt{b}) + p_{451}p(\mathtt{c}**)p(\mathtt{ab}*) + p_{452}p(*\mathtt{c}*)p(\mathtt{ab}*) + p_{453}p(**\mathtt{c})p(\mathtt{ab}*) \\
+ \;& p_{454}p(\mathtt{ab}*)\delta(\mathtt{a}=\mathtt{c}) + p_{455}p(\mathtt{ab}*)\delta(\mathtt{b}=\mathtt{c}) + p_{456}p(\mathtt{abc}) + p_{461}p(\mathtt{c}**)p(\mathtt{a}*\mathtt{b}) \\
+ \;& p_{462}p(*\mathtt{c}*)p(\mathtt{a}*\mathtt{b}) + p_{463}p(**\mathtt{c})p(\mathtt{a}*\mathtt{b}) + p_{464}p(\mathtt{a}*\mathtt{b})\delta(\mathtt{a}=\mathtt{c}) + p_{465}p(\mathtt{acb}) \\
+ \;& p_{466}p(\mathtt{a}*\mathtt{b})\delta(\mathtt{b}=\mathtt{c}) + p_{511}p(\mathtt{b}**)p(*\mathtt{a}*)\delta(\mathtt{b}=\mathtt{c}) + p_{512}p(\mathtt{bc}*)p(*\mathtt{a}*) + p_{513}p(\mathtt{b}*\mathtt{c})p(*\mathtt{a}*) \\
+ \;& p_{514}p(\mathtt{b}**)p(\mathtt{ca}*) + p_{515}p(\mathtt{b}**)p(*\mathtt{a}*)\delta(\mathtt{a}=\mathtt{c}) + p_{516}p(\mathtt{b}**)p(*\mathtt{ac}) + p_{521}p(\mathtt{cb}*)p(*\mathtt{a}*) \\
+ \;& p_{522}p(*\mathtt{b}*)p(*\mathtt{a}*)\delta(\mathtt{b}=\mathtt{c}) + p_{523}p(*\mathtt{bc})p(*\mathtt{a}*) + p_{524}p(*\mathtt{b}*)p(\mathtt{ca}*) + p_{525}p(*\mathtt{b}*)p(*\mathtt{a}*)\delta(\mathtt{a}=\mathtt{c}) \\
+ \;& p_{526}p(*\mathtt{b}*)p(*\mathtt{ac}) + p_{531}p(\mathtt{c}*\mathtt{b})p(*\mathtt{a}*) + p_{532}p(*\mathtt{cb})p(*\mathtt{a}*) + p_{533}p(**\mathtt{b})p(*\mathtt{a}*)\delta(\mathtt{b}=\mathtt{c}) \\
+ \;& p_{534}p(**\mathtt{b})p(\mathtt{ca}*) + p_{535}p(**\mathtt{b})p(*\mathtt{a}*)\delta(\mathtt{a}=\mathtt{c}) + p_{536}p(**\mathtt{b})p(*\mathtt{ac}) + p_{541}p(\mathtt{c}**)p(\mathtt{ba}*) \\
+ \;& p_{542}p(*\mathtt{c}*)p(\mathtt{ba}*) + p_{543}p(**\mathtt{c})p(\mathtt{ba}*) + p_{544}p(\mathtt{ba}*)\delta(\mathtt{b}=\mathtt{c}) + p_{545}p(\mathtt{ba}*)\delta(\mathtt{a}=\mathtt{c}) \\
+ \;& p_{546}p(\mathtt{bac}) + p_{551}p(\mathtt{c}**)p(*\mathtt{a}*)\delta(\mathtt{a}=\mathtt{b}) + p_{552}p(*\mathtt{c}*)p(*\mathtt{a}*)\delta(\mathtt{a}=\mathtt{b}) + p_{553}p(**\mathtt{c})p(*\mathtt{a}*)\delta(\mathtt{a}=\mathtt{b}) \\
+ \;& p_{554}p(\mathtt{ca}*)\delta(\mathtt{a}=\mathtt{b}) + p_{555}p(*\mathtt{a}*)\delta(\mathtt{a}=\mathtt{b})\delta(\mathtt{a}=\mathtt{c}) + p_{556}p(*\mathtt{ac})\delta(\mathtt{a}=\mathtt{b}) + p_{561}p(\mathtt{c}**)p(*\mathtt{ab}) \\
+ \;& p_{562}p(*\mathtt{c}*)p(*\mathtt{ab}) + p_{563}p(**\mathtt{c})p(*\mathtt{ab}) + p_{564}p(\mathtt{cab}) + p_{565}p(*\mathtt{ab})\delta(\mathtt{a}=\mathtt{c}) \\
+ \;& p_{566}p(*\mathtt{ab})\delta(\mathtt{b}=\mathtt{c}) + p_{611}p(\mathtt{b}**)p(**\mathtt{a})\delta(\mathtt{b}=\mathtt{c}) + p_{612}p(\mathtt{bc}*)p(**\mathtt{a}) + p_{613}p(\mathtt{b}*\mathtt{c})p(**\mathtt{a}) \\
+ \;& p_{614}p(\mathtt{b}**)p(\mathtt{c}*\mathtt{a}) + p_{615}p(\mathtt{b}**)p(*\mathtt{ca}) + p_{616}p(\mathtt{b}**)p(**\mathtt{a})\delta(\mathtt{a}=\mathtt{c}) + p_{621}p(\mathtt{cb}*)p(**\mathtt{a}) \\
+ \;& p_{622}p(*\mathtt{b}*)p(**\mathtt{a})\delta(\mathtt{b}=\mathtt{c}) + p_{623}p(*\mathtt{bc})p(**\mathtt{a}) + p_{624}p(*\mathtt{b}*)p(\mathtt{c}*\mathtt{a}) + p_{625}p(*\mathtt{b}*)p(*\mathtt{ca}) \\
+ \;& p_{626}p(*\mathtt{b}*)p(**\mathtt{a})\delta(\mathtt{a}=\mathtt{c}) + p_{631}p(\mathtt{c}*\mathtt{b})p(**\mathtt{a}) + p_{632}p(*\mathtt{cb})p(**\mathtt{a}) + p_{633}p(**\mathtt{b})p(**\mathtt{a})\delta(\mathtt{b}=\mathtt{c}) \\
+ \;& p_{634}p(**\mathtt{b})p(\mathtt{c}*\mathtt{a}) + p_{635}p(**\mathtt{b})p(*\mathtt{ca}) + p_{636}p(**\mathtt{b})p(**\mathtt{a})\delta(\mathtt{a}=\mathtt{c}) + p_{641}p(\mathtt{c}**)p(\mathtt{b}*\mathtt{a}) \\
+ \;& p_{642}p(*\mathtt{c}*)p(\mathtt{b}*\mathtt{a}) + p_{643}p(**\mathtt{c})p(\mathtt{b}*\mathtt{a}) + p_{644}p(\mathtt{b}*\mathtt{a})\delta(\mathtt{b}=\mathtt{c}) + p_{645}p(\mathtt{bca}) \\
+ \;& p_{646}p(\mathtt{b}*\mathtt{a})\delta(\mathtt{a}=\mathtt{c}) + p_{651}p(\mathtt{c}**)p(*\mathtt{ba}) + p_{652}p(*\mathtt{c}*)p(*\mathtt{ba}) + p_{653}p(**\mathtt{c})p(*\mathtt{ba}) \\
+ \;& p_{654}p(\mathtt{cba}) + p_{655}p(*\mathtt{ba})\delta(\mathtt{b}=\mathtt{c}) + p_{656}p(*\mathtt{ba})\delta(\mathtt{a}=\mathtt{c}) + p_{661}p(\mathtt{c}**)p(**\mathtt{a})\delta(\mathtt{a}=\mathtt{b}) + p_{662}p(*\mathtt{c}*)p(**\mathtt{a})\delta(\mathtt{a}=\mathtt{b}) \\
+ \;& p_{663}p(**\mathtt{c})p(**\mathtt{a})\delta(\mathtt{a}=\mathtt{b}) + p_{664}p(\mathtt{c}*\mathtt{a})\delta(\mathtt{a}=\mathtt{b}) + p_{665}p(*\mathtt{ca})\delta(\mathtt{a}=\mathtt{b}) + p_{666}p(**\mathtt{a})\delta(\mathtt{a}=\mathtt{b})\delta(\mathtt{a}=\mathtt{c})
\end{aligned}
$$

Figure 3: Evolution equation for a 3-locus representation evolved under selection and generalised recombination.

5 Equations for order 1 schemata

Let us focus on the order 1 schemata $H_s^a = *^{s-1}a*^{\ell-s}$ where only one allele is specified. By coarse-graining on the recombination distribution, the schema evolution equations for these schemata transform into:

$$\begin{aligned} & E[\Phi(H_s^a, t+1)] \\ &= \sum_{(m_s,v_s)\in\mathcal{R}_\ell} p_c(*^{s-1}m_s*^{\ell-s}, *^{s-1}v_s*^{\ell-s}) p\left(H_{v_s}^a, t\right) \\ &= \sum_{k=1}^{\ell} p_c(*\cdots*, *^{s-1}k*^{\ell-s}) p\left(H_k^a, t\right). \end{aligned}$$

That is, the evolution of order 1 schemata is governed by systems of ℓ linear equations. There are as many such systems as the arity of the alphabet adopted for strings. In the binary case $a \in \{0, 1\}$ and so there are two such systems.

So, in general, unlike the case for homologous crossovers, with generalised recombination, order 1 schemata may evolve even on a flat landscape (where $p(H, t) = \Phi(H, t)$ for any schema H). The flat landscape case is interesting as its analysis unveils the biases of the recombination operator. For the case $\ell = 2$ in [18] we found that, except in special conditions, a fixed point for the proportions of order 1 schemata $\Phi(H_s^a, t)$ exists. This is generally the case for any ℓ. Let us denote such a fixed point with $\Phi^*(H_s^a)$.

Let us consider the case of an infinitely large population and a flat landscape.[3] In vector notation then the system of equations becomes

$$\vec{\Phi}^a(t+1) = A\vec{\Phi}^a(t)$$

where $\vec{\Phi}^a(t) = [\Phi(H_1^a, t), \cdots, \Phi(H_\ell^a, t)]^T$ and $A = (a_{sk})$ is a matrix with elements $a_{sk} = p_c(*\cdots*, *^{s-1}k*^{\ell-s})$. Since $\sum_{k=1}^{\ell} p_c(*\cdots*, *^{s-1}k*^{\ell-s}) = p_c(*\cdots*, *\cdots*) = 1$ the matrix A is row stochastic, but it is not necessarily column stochastic.

5.1 Fixed points

Let us look for fixed points for the dynamical system defined by these equations. They will have to be eigenvectors of the matrix A with an associated eigenvalue $\lambda = 1$.

Because of the row stochasticity of A, it is easy to see that $[1, \cdots, 1]^T$ is an eigenvector for the matrix. That is, for order 1 schemata, a fixed point always exists of the form

$$\Phi^*(H_s^a) = c(a)$$

for $s = 1, \cdots, \ell$, where $c(a)$ is a constant (possibly a different one for each a). Naturally the constants $c(a)$ must obey the conservation of probability for the ℓ sets of order 1 schemata partitioning the search space. That is, we require that, for all s and t,

$$\sum_a \Phi(H_s^a, t) = 1.$$

When evaluated at the fixed point, this leads to the following constraint on the values of the $c(a)$'s:

$$\sum_a c(a) = 1.$$

Generally, finding analytically other fixed points may not be simple. Also, determining whether a fixed point is a global attractor for the system is non-trivial.[4] There are, however, some fairly general classes of generalised recombinations where we can say a bit more.

[3] Infinitely large populations are a standard mathematical tool in the theory of evolutionary algorithms. They are used because they remove the stochasticity present in EAs. This can be very useful, for example, to aid the analysis of the intrinsic biases of the search operators.

[4] Naturally, if the GRD is known, one can easily find *numerical* answers to these questions simply by using standard linear algebra techniques.

5.1.1 Homologous crossover

One such class is the class of homologous crossovers. These are characterised by the fact that only recombination pairs of the form $r = (b, (1, 2, \cdots, \ell))$ have non-zero probability. So, $a_{sk} = p_c(* \cdots *, *^{s-1} k *^{\ell - s}) = \delta(s = k)$ and, so, A is the identity matrix. In this case, as expected, any initial condition is a fixed point for order 1 schemata. That is

$$\Phi^*(H_s^a) = \Phi(H_s^a, 0).$$

5.1.2 Fully disconnected recombination cliques

Let $Q(p_c)$ the set of recombination cliques induced by the generalised recombination distribution p_c. The elements of $Q(p_c)$ are (disjoint) sets of integers. Their union is $\{1, \cdots, \ell\}$.

The homologous crossover case is a special case in which the recombination clique graph includes ℓ disconnected nodes (i.e., $|Q(p_c)| = \ell$). The fully mixing case is one where all nodes belong to a single clique (i.e., $|Q(p_c)| = 1$). Let us consider what happens in other cases similar as these, where the loci can be grouped into a number of cliques, but where the cliques themselves are completely disconnected. In other words, we consider the case where the recombination clique DAG includes $q = |Q(p_c)|$ nodes with $1 < q < \ell$ and *no arcs*.

In this case the matrix A is block diagonal, with q blocks. So, effectively we can decompose the vector $\vec{\Phi}^a$ into q sub-vectors $\vec{\Phi}_n^a$ and the matrix A into q squared sub-matrices A_n (the blocks along the diagonal of A) and rewrite the evolution equations for order 1 schemata as:

$$\vec{\Phi}_n^a(t+1) = A_n \vec{\Phi}_n^a(t)$$

for $n \in Q(p_c)$. It is then easy to see that each of these smaller dynamical systems has an eigenvalue $\lambda_n = 1$ with an associated eigenvector of the form $[1, \cdots, 1]^T$. So, a fixed point exists of the form

$$\vec{\Phi}_n^{a^*} = c(n, a)[1, \cdots, 1]^T$$

for $n \in Q(p_c)$, where $c(n, a)$ are constants which depend only on the clique n and the allele a. These, again, must respect the conservation of probability and so

$$\sum_a c(n, a) = 1.$$

6 Fixed points for higher-order schemata and strings

Let us consider the case where $p_c(m, v) = 0$ for all v such that $\exists i \neq j, v_i = v_j$, that is let us assume no allele duplication from the same parent can take place. We call this a *δ-free recombination distribution* because in these conditions all the δ's in Equation 4 (and the corresponding equation for $\bar{I}_r$) are all 1 for any r.

Theorem (Generalised Geiringer manifold) *A fixed point distribution for the proportion of a string or a schema $h_1 h_2 \cdots h_\ell$ under generalised crossover with a δ-free recombination distribution for an infinite population operating on a flat fitness landscape is given by*

$$\Phi^*(h_1 \cdots h_\ell) = \prod_{q \in Q(p_c)} \prod_{i \in q} c(q, h_i) \tag{5}$$

where $c(q, *) = 1$.

Proof Since the fitness landscape is flat, $p(H,t) = \Phi(H,t)$ for any schema. Also, because the population is infinite, $E[\Phi(H,t+1)] = \Phi(H,t+1)$. Then, for a δ-free GRD we can rewrite the schema evolution equations as

$$\begin{aligned}
&\Phi(h,t+1) \\
&= \sum_{(m,v)\in\mathcal{R}_\ell^\ell} p_c(m,v) \\
&\quad \Phi\left(\prod_{k=1}^{|I_r|}\left(*^{v_{i_k}-v_{i_{k-1}}-1}h_{i_k}\right)*^{\ell-v_{i_{|I_r|}}},t\right) \\
&\quad \Phi\left(\prod_{k=1}^{|\bar{I}_r|}\left(*^{v_{j_k}-v_{j_{k-1}}-1}h_{j_k}\right)*^{\ell-v_{j_{|\bar{I}_r|}}},t\right).
\end{aligned} \tag{6}$$

We can prove that Equation 5 is a fixed point for this equation, by substituting the right-hand side of Equation 5 into the right-hand side of this equation and then showing that the resulting expression for $\Phi(h_1...h_\ell, t+1)$ has exactly the same form as the right-hand side of Equation 5.

Let us start by splitting each I_r into disjoint subsets I_{rn} for $n \in Q(p_c)$ where subset I_{rn} includes the elements of I_r from clique n. That is $I_{rn} = I_r \cap n$. Then at the fixed point

$$\begin{aligned}
&\Phi\left(\prod_{k=1}^{|I_r|}\left(*^{v_{i_k}-v_{i_{k-1}}-1}h_{i_k}\right)*^{\ell-v_{i_{|I_r|}}},t\right) \\
&\quad = \prod_{n\in Q(p_c)}\prod_{i\in I_{rn}} c(n,h_i).
\end{aligned}$$

A similar result holds for $\bar{I}_r$ and the last term of Equation 6.

So, from the substitution of the fixed point in Equation 6 we obtain

$$\begin{aligned}
&\Phi(h,t+1) = \\
&\quad \sum_{r\in\mathcal{R}_\ell^\ell} p_c(r) \prod_{n\in Q(p_c)}\prod_{i\in I_{rn}} c(n,h_i) \prod_{n\in Q(p_c)}\prod_{j\in \bar{I}_{rn}} c(n,h_j)
\end{aligned}$$

Because I_r and $\bar{I}_r$ are disjoint and their union is $\{1,\cdots,\ell\}$, for all $n \in Q(p_c)$ we have $I_{rn} \cup \bar{I}_{rn} = n$ and, so,

$$\begin{aligned}
\Phi(h,t+1) &= \sum_{r\in\mathcal{R}_\ell^\ell} p_c(r) \prod_{n\in Q(p_c)}\prod_{i\in n} c(n,h_i) \\
&= \prod_{n\in Q(p_c)}\prod_{i\in n} c(n,h_i) \underbrace{\sum_{r\in\mathcal{R}_\ell^\ell} p_c(r)}_{=1} \\
&= \prod_{n\in Q(p_c)}\prod_{i\in n} c(n,h_i).
\end{aligned}$$

which proves that Equation 5 is a fixed point for the distribution of strings and, more generally, schemata. □

This result is important because *it provides a generalisation of the manifold described, for homologous crossover, by Geiringer [2].* All points on our generalised Geiringer manifold are fixed points for a genetic system under generalised recombination. Naturally, the result also covers all the fixed points for order one schemata described in the previous section.

It is interesting to rewrite Equation 5 is a slightly different form. If $\nu(h,n,a)$ represents the number of times symbol a appears in one of the loci in clique n of the string or schema h, and Ω represents our alphabet, then

$$\Phi^*(h) = \prod_{n \in Q(p_c)} \prod_{a \in \Omega} (c(n,a))^{\nu(h,n,a)}. \tag{7}$$

So, for example if our alphabet is $\Omega = \{0,1,2,3\}$, if $|Q(p_c)| = 1$ and if we set $c(n,0) = c(n,1) = 1/3$ and $c(n,2) = c(n,3) = 1/6$, then $\Phi^*(0102) = (1/2)^2 \times (1/2) \times (1/3) \times (1/3)^0 = 1/24$. Interestingly, in the case of a binary alphabet, for a fixed $c(n,0)$ (note: $c(n,1) = 1 - c(n,0)$) the probability of sampling a given string is only a function of the unitation value (the number of ones) of the string.

7 Stability of fixed points

Naturally, although any choice of $c(n,a)$ will provide a formal fixed point for the evolution equations, we are only interested in choices which respect the conservation of probability constraint $\sum_a c(n,a) = 1$. Despite this constraint, we still have a huge family of potential fixed points. An important question is whether any of these fixed points would be a global attractor for the system and whether this would dependent on initial conditions and, if so, how.

In this paper we don't formally prove under which conditions the fixed point presented in the previous sections are stable. In [18] we present an exact and general solution for the dynamics for the case $\ell = 2$ and a complete analysis of the corresponding fixed points. The techniques used there can provide exact answers also for $\ell > 2$. However, the complexity of the solutions grows very quickly with ℓ. So, in this paper we prefer to present empirical evidence to corroborate our theoretical results.

8 "Schemulator" runs

In order to study the dynamics of a genetic system under selection and generalised recombination we have implemented a simulator written in Java (we call it the "*schemulator*" – a contraction of "schema simulator") which expands and then numerically integrates the string (and schema) evolution equations for any choice of recombination distribution, of fitness function and of initial conditions. The integration is performed under the standard assumption of infinite populations.

To corroborate our results we want to verify our predictions as to the existence and location of fixed points for the flat fitness landscape case. Figure 4 shows the dynamics of some schemata and strings in a population with $\ell = 3$ and a recombination distribution where $p_c(m,v) \neq 0$ for all the 48 recombination pairs where v is a permutation vector, and $p_c(m,v) = 0$ for the remaining 168 pairs. The non-zero entries of the GRD were randomly generated and then normalised so that $\sum p_c(r) = 1$. The resulting recombination distribution had only one clique, $\mathcal{N}_\ell = \{1, \cdots, \ell\}$, which includes all ℓ loci. In order to be able to distinguish between the dynamics of different schemata, we the used unequal initial proportions for strings, namely: $\Phi(\mathtt{000}, 0) = 0.3$, $\Phi(\mathtt{001}, 0) = 0.25$, $\Phi(\mathtt{010}, 0) = \Phi(\mathtt{011}, 0) = \Phi(\mathtt{100}, 0) = 0.1$, $\Phi(\mathtt{101}, 0) = 0.05$, $\Phi(\mathtt{110}, 0) = 0.02$ and $\Phi(\mathtt{111}, 0) = 0.08$.

As shown in the figure, the order 1 schemata H_s^1 $(s = 1,2,3)$ rapidly converge to a fixed point where $\Phi^*(1**) = \Phi^*(*1*) = \Phi^*(**1)$. This is exactly what is predicted by the fixed point provided in Equation 5. The order-one-schema fixed point proportion, 0.343333333333, suggests that $c(\mathcal{N}_\ell, 1) = 0.343333333333$ and $c(\mathcal{N}_\ell, 0) = 1 - c(\mathcal{N}_\ell, 1) = 0.656666666667$.

Order 2 schemata also converge to identical values, i.e. $\Phi^*(11*) = \Phi^*(*11) = \Phi^*(1*1)$. The fixed-point frequency is (within numerical errors) exactly $c(\mathcal{N}_\ell, 1)^2 = 0.117877777778$, which is what Equation 5 predicts.

The predictions of our generalised Geiringer manifold theorem also hold for strings. For example, the strings 110 and 011 converge to their predicted fixed point $\Phi^*(110) = \Phi^*(011) = c(\mathcal{N}_\ell, 1)^2 c(\mathcal{N}_\ell, 0) = 0.0774064074076$ and 111 converges towards the predicted $\Phi^*(111) = c(\mathcal{N}_\ell, 1)^3 = 0.0404713703703$ within numerical errors.

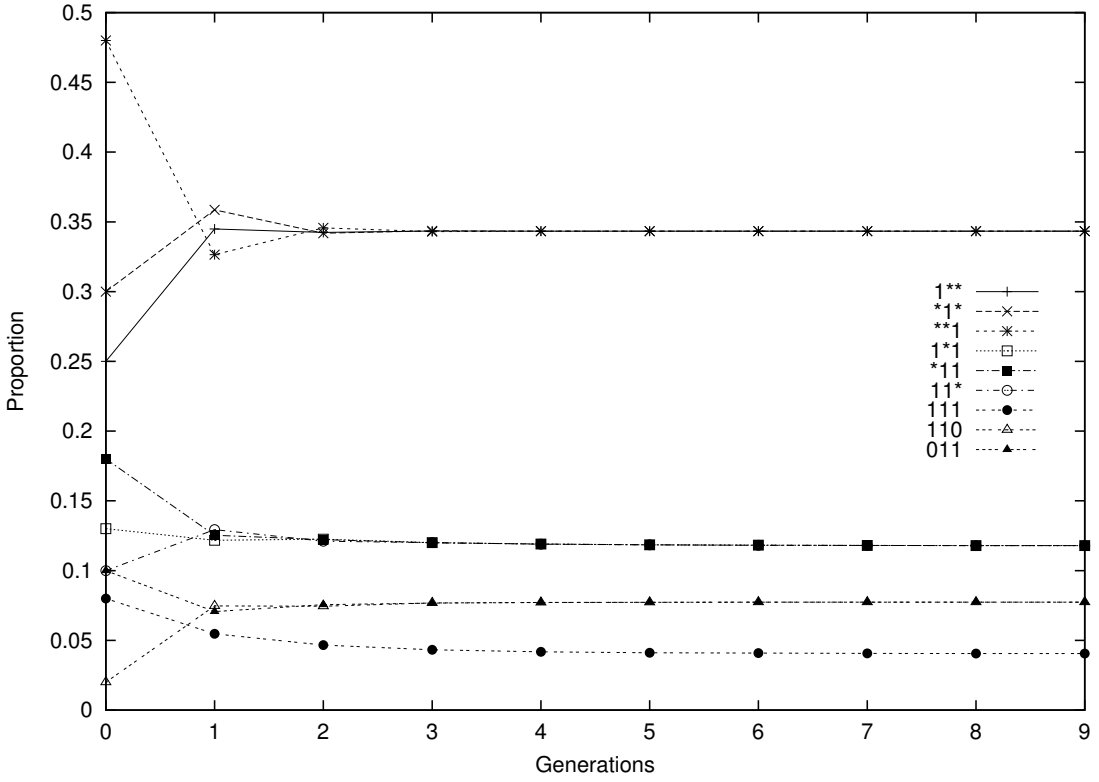

Figure 4: Dynamics of strings and schemata for $\ell = 3$ and a δ-free, fully mixing, random recombination distribution.

One might ask at this stage: where did the magic value 0.343333333333 come from? We provide here an answer without proof. The GRD used in this example is one of a class for which

$$c(n, a) = \frac{1}{|n|} \sum_{i \in n} \Phi(H_i^a, 0). \tag{8}$$

Since in this particular example we only have one clique,

$$\begin{aligned} c(\mathcal{N}_\ell, 1) &= \frac{1}{3} \left(\Phi(1**, 0) + \Phi(*1*, 0) + \Phi(**1, 0) \right) \\ &= \frac{1}{3} \left(0.25 + 0.3 + 0.48 \right) \\ &= 0.343333333333. \end{aligned}$$

9 Conclusions

In this paper we have provided a theory that is powerful enough to model exactly genetic systems using a fixed-length representation, selection and, for the first time, a rich set of genetic operations, including gene duplication, gene deletion, inversion, homologous recombination, permutations, diploidy, etc. that are not only known to happen in nature but that have also been fruitfully used in evolutionary algorithms. This model includes as a special case previous models such as the exact schema theory in [17, 14].

We have started analysing the evolution equations provided by our model with the objective of understanding the search biases induced by such a powerful set of operators. This has allowed us to

formulate a generalisation of Geiringer's theorem. As usual, we expect the study of the equations in the presence of selection to be much harder to do mathematically. However, the availability of an exact probabilistic model has allowed the implementation of an evolution equation simulator (the schemulator) with which we can numerically explore the interaction between the recombination and the selection biases for arbitrary fitness functions and potentially for any string length.

In future research we intend to provide a detailed general analysis of fixed-point stability, to study the evolution equations for diploid recombination distributions and to extend the results presented in this paper to the case of variable length strings, thereby, hopefully, contributing new results to theoretical population genetics as well as evolutionary computation.

Acknowledgements

CRS and RP thank Bill Langdon for helpful comments and ESPRC for financial support (grant number GR/T24616/01). CRS also thanks DGAPA of the UNAM for a Sabbatical Fellowship and Conacyt project 30422-E.

References

[1] A.G. Clark. Invasion and maintenance of a gene duplication. *Proc. Nat. Acad. Sci.*, 91:2950–2954, 1994.

[2] Hilda Geiringer. On the probability theory of linkage in Mendelian heredity. *Annals of Mathematical Statistics*, 15(1):25–57, March 1944.

[3] David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, Massachusetts, 1989.

[4] J. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, USA, 1975.

[5] John R. Koza. Gene duplication to enable genetic programming to concurrently evolve both the architecture and work-performing steps of a computer program. In *IJCAI-95 Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, volume 1, pages 734–740, Montreal, Quebec, Canada, 20-25 August 1995. Morgan Kaufmann.

[6] Nicholas Freitag McPhee and Riccardo Poli. Using schema theory to explore interactions of multiple operators. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002)*, New York, USA, July 2002. Morgan Kaufmann. (accepted as full paper).

[7] Riccardo Poli. Exact schema theory for genetic programming and variable-length genetic algorithms with one-point crossover. *Genetic Programming and Evolvable Machines*, 2(2):123–163, June 2001.

[8] Riccardo Poli. A simple but theoretically-motivated method to control bloat in genetic programming. In Conor Ryan, Terrence Soule, Maarten Keijzer, Edward Tsang, Riccardo Poli, and Ernesto Costa, editors, *Genetic Programming, Proceedings of the 6th European Conference, EuroGP 2003*, LNCS, pages 211–223, Essex, UK, 14-16 April 2003. Springer-Verlag.

[9] Riccardo Poli and Nicholas Freitag McPhee. General schema theory for genetic programming with subtree-swapping crossover: Part I. *Evolutionary Computation*, 11(1):53–66, 2003.

[10] Riccardo Poli and Nicholas Freitag McPhee. General schema theory for genetic programming with subtree-swapping crossover: Part II. *Evolutionary Computation*, 11(2), 2003.

[11] Mark Ridley. *Evolution*. Blackwell Scientific Publications, Boston, 1993.

[12] Jonathan E. Rowe, Michael D. Vose, and Alden H. Wright. Group properties of crossover and mutation. *Evolutionary Computation*, 10(2):151–184, 2002.

[13] Hidefumi Sawai and Susumu Adachi. A comparative study of gene-duplicated GAs based on pfGA and SSGA. In Darrell Whitley, David Goldberg, Erick Cantu-Paz, Lee Spector, Ian Parmee, and Hans-Georg Beyer, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)*, pages 74–81, Las Vegas, Nevada, USA, 10-12 July 2000. Morgan Kaufmann.

[14] C. R. Stephens. Some exact results from a coarse grained formulation of genetic dynamics. In Lee Spector, Erik D. Goodman, Annie Wu, W. B. Langdon, Hans-Michael Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max H. Garzon, and Edmund Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 631–638, San Francisco, California, USA, 7-11 July 2001. Morgan Kaufmann.

[15] C. R. Stephens and J. Mora Vargas. Effective fitness as an alternative paradigm for evolutionary computation I: General formalism. *Genetic Programming and Evolvable Machines*, 1(4):363–378, October 2000.

[16] C. R. Stephens and J. Mora Vargas. Effective fitness as an alternative paradigm for evolutionary computation II: Examples and applications. *Genetic Programming and Evolvable Machines*, 2(1):7–32, March 2001.

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

[18] Christopher R. Stephens and Riccardo Poli. Coarse graining in an evolutionary algorithm with recombination, duplication and inversion. Technical Report CSM-427, Department of Computer Science, University of Essex, 2005.

[19] Michael D. Vose. *The simple genetic algorithm: Foundations and theory.* MIT Press, Cambridge, MA, 1999.

[20] D. Whitley. A genetic algorithm tutorial. *Statistics and Computing*, 4(2):65–85, 1994.

[21] Darrell Whitley. An executable model of a simple genetic algorithm. In Darrell Whitley, editor, *Foundations of Genetic Algorithms Workshop (FOGA-92)*, Vail, Colorado, July 1992.