Implicit Parallelism

Alden H. Wright[1], Michael D. Vose[2], and Jonathan E. Rowe[3]

[1] Dept. of Computer Science, University of Montana, Missoula, Montana 59812, USA
wright@cs.umt.edu

[2] Computer Science Department, University of Tennessee, Knoxville, TN 37996, USA
vose@cs.utk.edu

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

Abstract. This paper assumes a search space of fixed-length strings, where the size of the alphabet can vary from position to position. Structural crossover is mask-based crossover, and thus includes n-point and uniform crossover. Structural mutation is mutation that commutes with a group operation on the search space. This paper shows that structural crossover and mutation project naturally onto competing families of schemata. In other words, the effect of crossover and mutation on a set of string positions can be specified by considering only what happens at those positions and ignoring other positions. However, it is not possible to do this for proportional selection except when fitness is constant on each schema of the family. One can write down an equation which includes selection which generalizes the Holland Schema theorem. However, like the Schema theorem, this equation cannot be applied over multiple time steps without keeping track of the frequency of every string in the search space.

1 Introduction

This paper describes the remarkable properties of structural crossover and mutation with respect to families of competing schemata. Recall that a schema defines certain components as taking on fixed values, while the remaining components are free to vary. Two schemata are in the same family if they specify fixed values for the same set of components. They are competing if they specify different values at these components. Given a set of components, the set of all possible fixed values that can be assigned to them gives us a whole competing family of schemata.

Vose and Wright, 2001 showed that each schema corresponds to a vector in population space. These schema vectors make up the components of a generalized (and redundant) coordinate system for population space. As the genetic algorithm moves from point to point in population space, it might be said to "process" schemata in the sense that each point in population space determines the frequency of each schema. This processing of many schemata has traditionally been called "implicit parallelism". However, the components corresponding to nontrivial schemata families are not independently processed and thus any implication that the genetic algorithm gains processing leverage due to the processing of schemata is misguided.

The set of components corresponding to a competing family of schemata may be picked out by a binary mask, and we can view such a mask as being a projection onto a smaller search space (in which we ignore what happens at the other positions). The remarkable result that we will prove is that structural crossover and mutation project naturally onto these families. That is, we can specify the effect of crossover and mutation just on the set of positions under consideration, ignoring what happens at other positions. Because this result applies to all schemata families simultaneously, we dub it the "Implicit Parallelism Theorem". In doing so, we hope to eradicate the previous usage of this phrase (which did not apply to a theorem at all, but to a fundamental mistake) and rescue it for a more suitable usage.

The result that operators project naturally onto schemata families applies only to crossover and mutation. It would be nice if it also applied to selection, and this possibility is investigated. The conclusion, however, is that it cannot (except in the trivial case where fitness is a constant for each schema in a schema family). One can, of course, write down an equation which involves selection, as well as crossover and mutation. This equation is, in fact, more elegant and meaningful than Holland's original Schema Theorem, but it suffers from the same fundamental flaw. In order to compute the average fitness of a schema at a given generation, one needs to know all the details of the entire population at that generation. It is therefore impossible to project onto a schemata family and ignore what happens at the other components. One implication of this is that one cannot iterate the equation: like the Schema Theorem, it can be applied over more than one time-step only by keeping track of the frequency of every string in the search space.

2 Structural Search Spaces and Operators

We generalize Vose, 1999 by introducing a class of genetic operators associated with a certain subgroup structure (for more details, see Rowe et al., 2002). Suppose the search space forms a group $(\Omega, \oplus)$ which has nontrivial subgroups $A_0, \ldots, A_{\ell-1}$ such that for all i, j, x,

1. $\Omega = A_0 \oplus \ldots \oplus A_{\ell-1}$
2. $i \neq j \implies A_i \cap A_j = \{0\}$
3. $x \oplus A_i = A_i \oplus x$

Then Ω is the *internal direct sum* of the A_i (which are *normal* subgroups of Ω) and each element $x \in \Omega$ has a unique representation $x = x_0 \oplus \ldots \oplus x_{\ell-1}$ where $x_i \in A_i$. The map

$$x \longmapsto \langle x_0, \ldots, x_{\ell-1} \rangle$$

is an isomorphism between Ω and the product group $A_0 \times \ldots \times A_{\ell-1}$ (see Lang, 1993), where

$$\langle x_0, \ldots, x_{\ell-1} \rangle \oplus \langle y_0, \ldots, y_{\ell-1} \rangle = \langle x_0 \oplus y_0, \ldots, x_{\ell-1} \oplus y_{\ell-1} \rangle$$

and

$$\ominus \langle x_0, \ldots, x_{\ell-1} \rangle = \langle \ominus x_0, \ldots, \ominus x_{\ell-1} \rangle$$

(where $\ominus$ indicates the inverse of an element). In this case the search space is called *structural*. Assume for the remainder of this paper that Ω is structural, and identify $x \in \Omega$ with $\langle x_0, \ldots, x_{\ell-1} \rangle$.

As an example, consider a length ℓ string representation where the cardinality of the alphabet at string position j is c_j. The alphabet at position j can be identified with $\mathcal{Z}_{c_j}$ (the integers modulo c_j), and the $\oplus$ operator can be componentwise addition (modulo c_j at position j). Then Ω is isomorphic to the direct product $\mathcal{Z}_{c_0} \times \ldots \times \mathcal{Z}_{c_{\ell-1}}$. This example extends the situation considered in Koehler et al., 1997 where $c_0 = c_1 = \cdots = c_{\ell-1}$. The standard example of fixed-length binary strings is a special case in which $c_j = 2$ for all positions j.

A concrete example of the above is: $\ell = 2$, $c_0 = 3$, $c_1 = 2$, so Ω is isomorphic to $\mathcal{Z}_3 \times \mathcal{Z}_2$. When we write elements of Ω as strings, the standard practice of putting the least significant bit to the right is followed. Thus,

$$\Omega = \{00, 01, 10, 11, 20, 21\} = \{0, 1, 2, 3, 4, 5\}$$

The group operator works by applying addition modulo 3 to the left bit, and addition modulo 2 to the right bit. For example

$$21 \oplus 11 = 00$$

The element 00 is the identity.

The set $\mathcal{B}$ of *binary masks* corresponding to Ω is

$$\mathcal{B} = \{\langle b_0, \ldots, b_{\ell-1}\rangle : b_i \in \mathcal{Z}_2\}$$

where $\mathcal{Z}_2$ is the set of integers modulo 2. Note that $\mathcal{B}$ is an Abelian group under component-wise addition modulo 2. It is notationally convenient to let $\oplus$ also denote the group operation on $\mathcal{B}$; hence $\oplus$ is *polymorphic*.[1] Let $\otimes$ denote component-wise multiplication on $\mathcal{B}$, and let $\mathbf{1} \in \mathcal{B}$ be the identity element for $\otimes$. For $b \in \mathcal{B}$, define $\bar{b}$ by

$$\bar{b} = \mathbf{1} \oplus b$$

It is notationally convenient to extend $\otimes$ to a commutative operator acting also between elements $b \in \mathcal{B}$ and elements $x \in \Omega$ by

$$\langle b_0, \ldots, b_{\ell-1}\rangle \otimes \langle x_0, \ldots, x_{\ell-1}\rangle = \langle b_0 x_0, \ldots, b_{\ell-1} x_{\ell-1}\rangle$$

where $0x_i = 0 \in \Omega$ and $1x_i = x_i \in \Omega$. Here the right hand sides are elements of Ω; hence $\otimes$ is polymorphic.

It is easy to check that for all $x, y \in \Omega$ and $u, v \in \mathcal{B}$

$$\begin{aligned}
x &= x \otimes \mathbf{1} \\
\mathbf{1} &= u \oplus \bar{u} \\
0 &= u \oplus u \\
0 &= u \otimes \bar{u} \\
(x \oplus y) \otimes u &= (x \otimes u) \oplus (y \otimes u) \\
(x \otimes u) \oplus (y \otimes \bar{u}) &= (y \otimes \bar{u}) \oplus (x \otimes u) \\
(x \otimes u) \otimes v &= x \otimes (u \otimes v) \\
\ominus(u \otimes x) &= u \otimes (\ominus x)
\end{aligned}$$

[1] An operator is polymorphic when its definition depends upon the type of its arguments.

To simplify notation, $\otimes$ takes precedence over $\oplus$ by convention. If $b \in \mathcal{B}$ is a mask then $\#b$ denotes the number of ones it contains.

Let χ be a probability distribution over the set of binary masks,

$$\chi_b = \text{the probability of mask } b$$

Structural crossover with distribution χ applied to parents u and v corresponds to choosing binary mask b with probability χ_b and then producing the offspring $u \otimes b \oplus \bar{b} \otimes v$. The probability that parents $u, v \in \Omega$ have child k is therefore

$$r(u, v, k) = \sum_{b \in \mathcal{B}} \chi_b [u \otimes b \oplus \bar{b} \otimes v = k]$$

The corresponding crossover scheme $\mathcal{C}$ is also called structural and satisfies

$$\mathcal{C}(p)_k = \sum_{u,v} p_u p_v \sum_{b \in \mathcal{B}} \frac{\chi_b + \chi_{\bar{b}}}{2} [u \otimes b \oplus \bar{b} \otimes v = k]$$

where $p \in \mathbb{R}^{|\Omega|}$ is a distribution over the search space Ω. That is, p is a *population vector* in which p_k is the proportion of the population made up of copies of element k. $\mathcal{C}(p)$ gives the expected distribution after the application of crossover. For example, for uniform crossover with crossover rate u, the probability distribution χ is given by $\chi_0 = 1 - u + u/2^\ell$ and $\chi_b = u/2^\ell$ for $b \neq 0$.

Let μ be a probability distribution over Ω,

$$\mu_k = \text{the probability of } k$$

Structural mutation with distribution μ applied to $v \in \Omega$ corresponds to choosing k with probability μ_k and then producing the result $v \oplus k$. The probability that v mutates to u is therefore

$$U_{u,v} = \mu_{\ominus v \oplus u}$$

The corresponding mutation scheme $\mathcal{U}$ is also called structural. $\mathcal{U}(p) = Up$ gives the effect of applying mutation to population vector $p \in \mathbb{R}^{|\Omega|}$.

3 Masks as Projections

This section generalizes Vose and Wright, 2001. Assume Ω is structural, crossover is structural, and mutation is structural. Each binary mask b has associated subgroup

$$\Omega_b = b \otimes \Omega$$

The map

$$x \longmapsto b \otimes x$$

is a homomorphism from Ω to Ω_b since $b \otimes (x \oplus y) = b \otimes x \oplus b \otimes y$. The kernel is the normal subgroup $\Omega_{\bar{b}}$, and, therefore, the following map from the image Ω_b to the quotient group $\Omega / \Omega_{\bar{b}}$ is an isomorphism Lang, 1993,

$$z = z \otimes b \longmapsto \Omega_{\bar{b}} \oplus z$$

The quotient group $\Omega/\Omega_{\bar{b}} = \{\Omega_{\bar{b}} \oplus z : z \in \Omega_b\}$, being comprised of disjoint schemata, is referred to as the *schema family corresponding to* b, and schema $\Omega_{\bar{b}} \oplus z$ is referred to as the *schema corresponding to* $z \in \Omega_b$.

For $b \in \mathcal{B}$, define Λ_b as

$$\Lambda_b = \left\{ p \in \mathbb{R}^{|\Omega_b|} : p_k \geq 0, \ \sum p_k = 1 \right\}$$

The linear operator $\Xi_b : \mathbb{R}^{|\Omega|} \longrightarrow \mathbb{R}^{|\Omega_b|}$ with matrix

$$(\Xi_b)_{i,j} = [j \otimes b = i]$$

is called the *operator associated with the schema family corresponding to* b; it has rows indexed by elements of Ω_b and columns indexed by Ω. Notice that $\Xi_b(\Lambda) \subseteq \Lambda_b$. To simplify notation, we will refer simply to Ξ when the binary mask b is understood.

For the example of the fixed length string representation where Ω is isomorphic to $\mathcal{Z}_3 \times \mathcal{Z}_2$, for $b = 10$,

$$\Xi_{10} = \begin{bmatrix} 1\,1\,0\,0\,0\,0 \\ 0\,0\,1\,1\,0\,0 \\ 0\,0\,0\,0\,1\,1 \end{bmatrix}$$

and for $b = 01$,

$$\Xi_{01} = \begin{bmatrix} 1\,0\,1\,0\,1\,0 \\ 0\,1\,0\,1\,0\,1 \end{bmatrix}$$

Note that

$$\begin{aligned} \sum_i (\Xi p)_i &= \sum_{i \in \Omega_b} \sum_j [j \otimes b = i] p_j \\ &= \sum_j p_j \sum_{i \in \Omega_b} [j \otimes b = i] \\ &= \sum_j p_j \end{aligned}$$

Hence if $p \in \Lambda$ is a probability vector, then $\Xi p \in \Lambda_b$ is a probability vector. As the following computation shows, the i th component of Ξp is simply the proportion of the population p which is contained in the schema $\Omega_{\bar{b}} \oplus i$ which corresponds to $i \in \Omega_b$,

$$\begin{aligned} (\Xi p)_i &= \sum_j [j \otimes b = i] p_j \\ &= \sum_j [j \otimes \bar{b} \oplus j \otimes b = j \otimes \bar{b} \oplus i] p_j \\ &= \sum_j [j = j \otimes \bar{b} \oplus i] p_j \\ &\leq \sum_j [j \in \Omega_{\bar{b}} \oplus i] p_j \end{aligned}$$

Conversely, given $i \in \Omega_b$,

$$\begin{aligned}\sum_j [j \in \Omega_{\bar{b}} \oplus i] p_j &\le \sum_j [b \otimes j \in b \otimes \Omega_{\bar{b}} \oplus b \otimes i] p_j \\ &= \sum_j [j \otimes b = i] p_j\end{aligned}$$

The matrix Ξ therefore projects from the distribution over all possible strings to a distribution over a family of competing schemata. For example, using traditional schema notation, we could write:

$$\Xi_{10}(p_{00}, p_{01}, p_{10}, p_{11}, p_{20}, p_{21}) = (p_{0\,*}, p_{1\,*}, p_{2\,*})$$

and

$$\Xi_{01}(p_{00}, p_{01}, p_{10}, p_{11}, p_{20}, p_{21}) = (p_{*\,0}, p_{*\,1})$$

Let $\mathcal{B}_b = b \otimes \mathcal{B}$. It is notationally convenient to make Ξ_b polymorphic by extending it to also represent the linear map $\Xi_b : \mathbb{R}^{|\mathcal{B}|} \longrightarrow \mathbb{R}^{|\mathcal{B}_b|}$ with matrix

$$(\Xi_b)_{i,j} = [j \otimes b = i]$$

Here the rows are indexed by elements of $\mathcal{B}_b$ and columns are indexed by $\mathcal{B}$. Again, we will drop the subscript and refer simply to Ξ when the mask is understood.

For the example of the fixed length string representation where Ω is isomorphic to $\mathcal{Z}_3 \times \mathcal{Z}_2$, the set of masks is $\mathcal{B} = \{00, 01, 10, 11\}$. For $b = 10$,

$$\Xi_{10} = \begin{bmatrix} 1\,1\,0\,0 \\ 0\,0\,1\,1 \end{bmatrix},$$

and for $b = 01$,

$$\Xi_{01} = \begin{bmatrix} 1\,0\,1\,0 \\ 0\,1\,0\,1 \end{bmatrix}.$$

Note that

$$\begin{aligned}\sum_i (\Xi x)_i &= \sum_{i \in \mathcal{B}_b} \sum_j [j \otimes b = i] x_j \\ &= \sum_j x_j \sum_{i \in \mathcal{B}_b} [j \otimes b = i] \\ &= \sum_j x_j\end{aligned}$$

Hence if $x \in \mathbb{R}^{|\mathcal{B}|}$ is a probability vector, then $\Xi x \in \mathbb{R}^{|\mathcal{B}_b|}$ is a probability vector.

4 The Implicit Parallelism Theorem

Given that $\Omega = A_0 \oplus \cdots \oplus A_{\ell-1}$ is structural, Ω_b is also structural,

$$\Omega_b = A_{k_0} \oplus \cdots \oplus A_{k_{\#b-1}}$$

where $\{k_0, \ldots, k_{\#b-1}\} = \{i : b_i = 1\}$. Moreover, Λ_b is precisely the Λ previously defined as corresponding to the search space, if the search space is chosen to be Ω_b. Likewise, $\mathcal{B}_b$ is precisely the $\mathcal{B}$ previously defined as corresponding to the search space, if the search space is chosen to be Ω_b. Therefore, since $\Xi\chi$ and $\Xi\mu$ are probability vectors indexed by $\mathcal{B}_b$ and Ω_b (respectively), they have corresponding structural crossover and mutation schemes $\mathcal{C}_b$ and $\mathcal{U}_b$ which represent crossover and mutation on the state space Λ_b.

Theorem 1. *If $\mathcal{M} = \mathcal{U} \circ \mathcal{C}$, then $\Xi\mathcal{M}(x) = \mathcal{U}_b \circ \mathcal{C}_b(\Xi x)$*

Proof Let $\mathcal{M}_b = \mathcal{U}_b \circ \mathcal{C}_b$. The k th component of $\Xi\mathcal{M}(x)$ is

$$\begin{aligned}
\sum_{k' \in \Omega_{\overline{b}} \oplus k} \mathcal{M}(x)_{k'} &= \sum_{k' \in \Omega_{\overline{b}}} \mathcal{M}(x)_{k \oplus k'} \\
&= \sum_{u,v \in \Omega_b} \sum_{u',v' \in \Omega_{\overline{b}}} x_{u \oplus u'} x_{v \oplus v'} \sum_{k' \in \Omega_{\overline{b}}} M_{u \oplus u' \ominus k \ominus k', v \oplus v' \ominus k \ominus k'} \\
&= \sum_{u,v \in \Omega_b} \sum_{u' \in \Omega_{\overline{b}}} x_{u \oplus k \oplus u'} \sum_{v' \in \Omega_{\overline{b}}} x_{v \oplus k \oplus v'} \sum_{k' \in \Omega_{\overline{b}}} M_{u \ominus k', v \oplus v' \ominus u' \ominus k'}
\end{aligned}$$

The innermost sum above is

$$\sum_{k' \in \Omega_{\overline{b}}} \sum_{i \in \Omega_b} \sum_{i' \in \Omega_{\overline{b}}} \sum_{j \in \mathcal{B}_b} \sum_{j' \in \otimes \mathcal{B}_{\overline{b}}} \mu_{i \oplus i'} \frac{\chi_{j \oplus j'} + \chi_{\overline{j \oplus j'}}}{2}$$
$$[(i \oplus i') \oplus (u \ominus k') \otimes (j \oplus j') \oplus (\overline{j \oplus j'}) \otimes (v \oplus v' \ominus u' \ominus k') = 0]$$

Note that the indicator function above is equivalent to

$$[i' \ominus k' \otimes j' \oplus (\overline{b} \oplus j') \otimes (v' \ominus u' \ominus k') = 0][i \oplus u \otimes j \oplus (b \oplus j) \otimes v = 0]$$

The first factor above is equivalent to $[i' \oplus (\overline{b} \oplus j') \otimes (v' \ominus u') = k']$ which determines k'. This is most easily seen by choosing the search space to be $\Omega_{\overline{b}}$, in which case the first factor is an expression over the search space and its binary masks, and $\overline{b}$ is the identity element for $\otimes$; in that context $(\overline{b} \oplus j')$ is $\overline{j}'$ and the first factor becomes

$$\begin{aligned}
&[i' \ominus k' \otimes j' \oplus \overline{j}' \otimes (v' \ominus u' \ominus k') = 0] \\
&= [i' \ominus k' \otimes j' \oplus \overline{j}' \otimes (v' \ominus u') \ominus \overline{j}' \otimes k' = 0] \\
&= [i' \oplus \overline{j}' \otimes (v' \ominus u') \ominus k' \otimes j' \ominus \overline{j}' \otimes k' = 0] \\
&= [i' \oplus \overline{j}' \otimes (v' \ominus u') \ominus k' = 0] \\
&= [i' \oplus \overline{j}' \otimes (v' \ominus u') = k']
\end{aligned}$$

It follows that the sum above is

$$\begin{aligned}
&\sum_{i \in \Omega_b} \sum_{j \in \mathcal{B}_b} [i \oplus u \otimes j \oplus (b \oplus j) \otimes v = 0] \sum_{i' \in \Omega_{\overline{b}}} \mu_{i \oplus i'} \sum_{j' \in \otimes \mathcal{B}_{\overline{b}}} \frac{\chi_{j \oplus j'} + \chi_{\overline{j \oplus j'}}}{2} \\
&= \sum_{i \in \Omega_b} \sum_{j \in \mathcal{B}_b} [i \oplus u \otimes j \oplus (b \oplus j) \otimes v = 0] \, (\varXi \mu)_i \, \frac{(\varXi \chi)_j + (\varXi \chi)_{\overline{j}}}{2} \\
&= (M_b)_{u,v}
\end{aligned}$$

Where M_b is the mixing matrix for $\mathcal{M}_b$. Therefore, the The kth component of $\varXi\mathcal{M}(x)$ is

$$\begin{aligned}
&\sum_{u,v \in \Omega_b} (M_b)_{u,v} \sum_{u' \in \Omega_{\overline{b}}} x_{u \oplus k \oplus u'} \sum_{v' \in \Omega_{\overline{b}}} x_{v \oplus k \oplus v'} \\
&= \sum_{u,v \in \Omega_b} (M_b)_{u,v} (\varXi x)_{u \oplus k} (\varXi x)_{v \oplus k} \\
&= \mathcal{M}_b(\varXi x)_k
\end{aligned}$$

□

Corollary 1 (Implicit Parallelism).

$$\mathcal{M} = \mathcal{U} \circ \mathcal{C} \Longrightarrow \varXi\mathcal{M} = \mathcal{M}_b \circ \varXi \;\; \textit{where } \mathcal{M}_b = \mathcal{U}_b \circ \mathcal{C}_b$$
$$\mathcal{M} = \mathcal{C} \circ \mathcal{U} \Longrightarrow \varXi\mathcal{M} = \mathcal{M}_b \circ \varXi \;\; \textit{where } \mathcal{M}_b = \mathcal{C}_b \circ \mathcal{U}_b$$

In particular, $\varXi\mathcal{U} = \mathcal{U}_b \circ \varXi$ *and* $\varXi\mathcal{C} = \mathcal{C}_b \circ \varXi$*.*

Proof The first implication is theorem 1. A special case is $\mathcal{C} = \mathcal{I}$, in which case the conclusion is

$$\varXi\mathcal{U}(x) = \mathcal{U}_b(\varXi x)$$

Another special case is $\mathcal{U} = \mathcal{I}$, in which case the conclusion is

$$\varXi\mathcal{C}(x) = \mathcal{C}_b(\varXi x)$$

Consequently,

$$\varXi\mathcal{C} \circ \mathcal{U} = \mathcal{C}_b \circ \varXi \circ \mathcal{U} = \mathcal{C}_b \circ \mathcal{U}_b \circ \varXi$$

□

Corollary 1 speaks to schemata through the isomorphism $\Omega_b \cong \Omega/\Omega_{\overline{b}}$ given by

$$x \longmapsto \Omega_{\overline{b}} \oplus x$$

Therefore, $\mathcal{M}_b$ represents mixing (i.e., crossover and mutation) on a search space of schemata (i.e., the schema family $\Omega/\Omega_{\overline{b}}$).

A consequence of corollary 1 is that, independent of the order of crossover and mutation, *the following commutative diagram holds, in parallel, for every choice of* *schema family, simultaneously*

$$\begin{tikzcd} x \arrow[r] \arrow[d, "\Xi"'] & \mathcal{M}(x) \arrow[d, "\Xi"] \\ \Xi x \arrow[r] & \mathcal{M}_b(\Xi x) \end{tikzcd}$$

Because this result does speak to parallelism and schemata—subjects which implicit parallelism has classically dealt with—Vose (1999) has redefined the phrase "implicit parallelism" to refer to it.[2] This use of the term conflicts with that employed by GA practitioners (for example in Holland, 1975,Goldberg, 1989). To the extent "implicit parallelism" has traditionally indicated that some kind of "processing leverage" is enjoyed by GAs, traditional usage has been misguided; genetic algorithms exhibit no such behaviour, nor can the theorems of Holland establish such a result. Because corollary 1 *does* address exactly what happens within all schema families, in parallel, simultaneously, it is proposed as an appropriate alternative to take over the "Implicit Parallelism" label, in the hope that the misguided and incorrect traditional notion be eradicated.

Example

Let us consider the implicit parallelism of mutation on the example $\Omega = \mathcal{Z}_3 \times \mathcal{Z}_2$. Firstly, let us define our mutation operator by the probability distribution:

$$\mu_j = \begin{cases} 0.9 & \text{if } j = 00 \\ 0.02 & \text{otherwise} \end{cases}$$

That is, there is a probability of 0.9 that no mutation will take place. Otherwise, we pick an element $j \in \Omega$ at random (uniformly) and apply it to our current individual. Now suppose that we are interested in what happens in the first component. That is, we are concerned with the effect of mutation on the family of schemata $0\,*, 1\,*, 2\,*$. One way to calculate this would be to work out the effect of mutation on the whole population and then sum up the results for each schema in the family. The implicit parallelism theorem tells us that we don't need to do this. Instead, we can find a mutation operator that acts on the family of schemata itself, and has the exact equivalent effect.

For a concrete example, consider the population vector

$$p = (p_{00}, p_{01}, p_{10}, p_{11}, p_{20}, p_{21}) = (0.1, 0.2, 0.1, 0.2, 0.25, 0.15)$$

Our family of schemata corresponds to the mask $b = 10$. We have already seen that this gives us a matrix

$$\Xi_{10} = \begin{bmatrix} 1\,1\,0\,0\,0\,0 \\ 0\,0\,1\,1\,0\,0 \\ 0\,0\,0\,0\,1\,1 \end{bmatrix}$$

[2] ... in the binary case. This paper establishes the result more generally.

Multiplying p by this matrix gives us the distribution of the population over the family of schemata:

$$\Xi_{10} p = (p_{0\,*}, p_{1\,*}, p_{2\,*}) = (0.3, 0.3, 0.4)$$

We now have to define a mutation operator for this reduced search space. This is given by

$$\Xi_{10} \mu = (0.92, 0.04, 0.04)$$

So our mutation operator acting on our family of schemata consists of picking an element of $\{0\,*, 1\,*, 2\,*\}$ according to the above probability distribution and applying it to the element to be mutated. Notice that in this quotient group the element $0\,*$ is the identity. Constructing the mutation operator that acts on Λ_b from this distribution gives us

$$\mathcal{U}_{10}(x) = \begin{bmatrix} 0.92 & 0.04 & 0.04 \\ 0.04 & 0.92 & 0.04 \\ 0.04 & 0.04 & 0.92 \end{bmatrix} x$$

So in our example, we calculate the effect of mutation on the family of schemata as being

$$\begin{bmatrix} 0.92 & 0.04 & 0.04 \\ 0.04 & 0.92 & 0.04 \\ 0.04 & 0.04 & 0.92 \end{bmatrix} \begin{bmatrix} 0.3 \\ 0.3 \\ 0.4 \end{bmatrix} = \begin{bmatrix} 0.304 \\ 0.304 \\ 0.392 \end{bmatrix}$$

Notice that to make this calculation we did not need to know the details of the population p. We only needed to know how many elements were in each schema (given by Ξp). We can check this result by working out the effect of mutation on the whole population and then summing over the schemata. The implicit parallelism theorem tells us that we will get exactly the same result.

5 Implicit Parallelism and Fitness-Based Selection

It would be especially useful if, in the commutative diagram above, $\mathcal{M}$ could be generalized to $\mathcal{G}$ so the effects of selection could be incorporated. For proportional selection at least, Vose has pointed out the difficulties involved and concluded that such commutativity is in general not possible Vose, 1999. In an attempt to *force* commutativity, a selection scheme $\mathcal{F}_b$ might be defined on the quotient by

$$\mathcal{F}_b(\Xi x) = \Xi \mathcal{F}(x)$$

The problem here is that $\mathcal{F}_b$ is not well defined; the right hand side might depend on the particular x involved even though the left hand side does not (i.e., even if Ξx does not). In an attempt to ignore this complication, one might define a "fitness vector" f_b (over Ω_b) for which

$$\mathcal{F}_b(\Xi x) = \frac{\text{diag}\,(f_b)\,\Xi x}{f_b^T\,\Xi x}$$

Since the complication *cannot* be ignored, the vector f_b *must depend on* x. If f_b is defined as

$$f_b = \text{diag}\,(\Xi x)^{-1}\,\Xi\,\text{diag}\,(f)\,x$$

then

$$\begin{aligned} f_b^T\,\Xi x &= \sum_i (\Xi x)_i^{-1} (\Xi\,\text{diag}\,(f)\,x)_i\,(\Xi x)_i \\ &= \sum_i (\Xi\,\text{diag}\,(f)\,x)_i \\ &= \sum_j (\text{diag}\,(f)\,x)_j \\ &= f^T x \end{aligned}$$

Therefore, by way of notational sleight of hand,

$$\begin{aligned} \mathcal{F}_b(\Xi x) &= \frac{\text{diag}\,(f_b)\,\Xi x}{f^T x} \\ &= \frac{\text{diag}\,(\text{diag}\,(\Xi x)^{-1}\Xi\,\text{diag}\,(f)\,x)\,\Xi x}{f^T x} \\ &= \frac{\Xi\,\text{diag}\,(f)\,x}{f^T x} \\ &= \Xi\mathcal{F}(x) \end{aligned}$$

Of course, this definition for f_b is precisely the one given in the "schema theorem" Holland, 1975. Using this definition, one could define

$$\mathcal{G}_b = \mathcal{M}_b \circ \mathcal{F}_b$$

and appeal to implicit parallelism to conclude

$$\Xi\mathcal{G}(x) = \Xi\mathcal{M} \circ \mathcal{F}(x) = \mathcal{M}_b \circ \Xi \circ \mathcal{F}(x) = \mathcal{M}_b \circ \mathcal{F}_b(\Xi x) = \mathcal{G}_b(\Xi x)$$

thereby "extending" implicit parallelism from $\mathcal{M}$ to $\mathcal{G}$. Unlike Holland's result, the relation $\Xi\mathcal{G}(x) = \mathcal{G}_b(\Xi x)$

- is an equality which in every case provides nonvacuous information,
- says something nontrivial about new elements produced by mixing,
- makes explicit the relationships between the genetic operators and the underlying group structure of the search space.

However, it should be noted that because f_b depends on x, the "extension"

$$\Xi\mathcal{G}(x) = \mathcal{G}_b(\Xi x)$$

speaks only to what happens over a single time step (like Holland's result) and the information provided is insufficient to characterize the next generation (even in Λ_b). In particular, it cannot be used to map out population trajectories, and it certainly cannot be used to justify talk of "above average fitness building blocks being selected exponentially". The "fitness" of schemata cannot be well-defined (it is not an attribute of schemata Ξx, but is determined instead by x). The various claims about GAs that are traditionally made under the name of the *building block hypothesis* have, to date, no basis in theory, and, in some cases, are simply incoherent. One exception is when the fitness function is a constant for each schema of a schema family (in which case the remaining constituents are redundant). Otherwise, one must take account of the fact that schemata "fitnesses" are *dynamic* quantities that change from population to population (see Stephens and Waelbroeck, 1999 for such a view of the building block hypothesis). Moreover, the fitness of such a "building block" at a given time depends on the entire microscopic structure of the population at that time.

Example

Consider the family of schemata $0*$, $1*$, $2*$ on the search space $\Omega = \mathcal{Z}_3 \times \mathcal{Z}_2$. Let the fitness vector be $f = (f_{00}, f_{01}, f_{10}, f_{11}, f_{20}, f_{21})$. Then f_b can be calculated as:

$$f_b = \begin{bmatrix} \frac{f_{00}x_{00}+f_{01}x_{01}}{x_{00}+x_{01}} \\ \frac{f_{10}x_{10}+f_{11}x_{11}}{x_{10}+x_{11}} \\ \frac{f_{20}x_{20}+f_{21}x_{21}}{x_{20}+x_{21}} \end{bmatrix}$$

We can verify that $\mathcal{F}_b(\Xi x) = \Xi\mathcal{F}(x)$.

$$\mathcal{F}_b(\Xi x) = \begin{bmatrix} \frac{f_{00}x_{00}+f_{01}x_{01}}{x_{00}+x_{01}} & 0 & 0 \\ 0 & \frac{f_{10}x_{10}+f_{11}x_{11}}{x_{10}+x_{11}} & 0 \\ 0 & 0 & \frac{f_{20}x_{20}+f_{21}x_{21}}{x_{20}+x_{21}} \end{bmatrix} \begin{bmatrix} x_{00}+x_{01} \\ x_{10}+x_{11} \\ x_{20}+x_{21} \end{bmatrix} = \Xi\mathcal{F}(x)$$

Notice that the fitness of a schema depends on the details of the whole population x and not just on the corresponding schemata family.

6 Conclusions

This paper has developed a framework for the theory of genetic algorithms that use a fixed-length string representation where the cardinality of the alphabet at each string position is arbitrary. Structural crossover and mutation represent the natural ways to define crossover and mutation in this framework. An implicit paralllelism is proved. This theorem states that structural crossover and mutation project naturally onto all competing families of schemata. This kind of projection does not work for proportional selection except when fitness is constant on each schema of the family. An exact equation which generalizes the Holland Schema theorem can be proved, but like the Holland Schema theorem, it cannot be applied in realistic situations for more than one time step.

References

Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization & Machine Learning*. Addison Wesley, Reading, MA.

Holland, J. (1975). *Adapdation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, Michigan.

Koehler, Bhattacharyya, and Vose (1997). General cardinality genetic algorithms. *Evolutionary Computation*, 5(4):439–459.

Lang, S. (1993). *Algebra*. Addison-Wesley, Reading, MA, third edition.

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

Stephens, C. and Waelbroeck, H. (1999). Schemata evolution and building blocks. *Evolutionary Computation*, 7(2).

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

Vose, M.D. and Wright, A.H. (2001). Form invariance and implicit parallelism. *Evolutionary Computation*, 9(3):355–370.