Representation Invariant Genetic Operators

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

Michael D. Vose vose@eecs.utk.edu
Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville, Tennessee, 37996-3450, USA

Alden H. Wright alden.wright@umontana.edu
Department of Computer Science, The University of Montana, Missoula, Montana 59812, USA

Abstract

A genetic algorithm is invariant with respect to a set of representations if it runs the same no matter which of the representations is used. We formalize this concept mathematically, showing that the representations generate a group that acts upon the search space. Invariant genetic operators are those that commute with this group action. We then consider the problem of characterizing crossover and mutation operators that have such invariance properties. In the case where the corresponding group action acts transitively on the search space, we provide a complete characterization, including high-level representation-independent algorithms implementing these operators.

Keywords

Representations, crossover, mutation, group action, invariance, symmetry, coarse-graining.

1 Introduction

Suppose two people set out to implement a genetic algorithm (GA) for a specific problem—an instance of the knapsack problem, say. They agree to use a generational GA, with the same population size. Solutions will be represented by binary strings to indicate which objects are in the knapsack, and which are excluded. They use the same mutation rate (with bitwise mutation) and the same crossover rate (one-point crossover). However, when they run their experiments, they get significantly different results. Assuming there are no bugs in the software, what could be the reason?

One possibility is that the bitstring representations order the objects differently. For example, one may order the objects by increasing weight, and the other by increasing value. The effect of one-point crossover will be different in each case. We say that one-point crossover is not *invariant* under changes in the order of objects.

The point of this story is that when we design representations and genetic operators for a particular problem, we are implicitly making certain commitments about what features of the representation are significant, and what can be ignored. When using one-point crossover, for example, the assumption is that the order matters, whereas for uniform crossover this is not the case.

In this paper, we begin by formalizing what it means for a genetic operator to be invariant with respect to a set of possible representations (Section 2). We illustrate this formalism in Section 3 with some examples: the bitstring and permutation examples already mentioned, and a new problem based on balanced partitions of a set of objects. We then ask the following general question: given a set of possible representations for a problem, how can we characterize the possible crossover and mutation operators that are invariant with respect to these representations? Given a certain technical assumption on the properties of this set (namely, that they generate a group that acts transitively on the search space) we provide a complete characterization of such operators in Section 4, including general representation-independent algorithms for their implementation.[1] This result is illustrated by again referring to our three examples in Section 5.

The proof of the characterization theorem is somewhat complicated. Details and proofs are postponed to Section 6. Those readers who have no interest in mathematics may therefore stop reading after Section 5, and still come away with a general recipe for the design of new crossover and mutation operators. For those who are interested, the proof is based around a rather interesting result that states that any GA that is invariant with respect to a set of representations (again subject to a technical condition) can be seen as the projection, or *coarse graining* of another GA running on a larger search space.

2 Representation Invariance

Given a particular combinatorial optimization problem, there is a (finite) set S, called the *solution space*, whose elements are candidate solutions to the problem. These elements are rated by an *objective function* $F : S \to \mathbb{R}$, which is to be optimized. This function may include a penalty function or other method of handling constraints. In order to implement a GA for the problem, we create a *search space*, which we denote by Ω, whose elements are the objects on which crossover and mutation will act. For example, if S is the set of all subsets of some larger set containing ℓ elements, then Ω could be the set of binary strings of length ℓ.

Definition 1: *A* representation *is a bijection from the solution space of a problem to the search space.*

That is, a representation tells us which of the elements of the search space will be used to represent which elements of the solution space. Continuing the subset example, one possible representation (call it a_1) would order the objects of the larger set in some way, and then use a 1 to indicate the presence of an object in the subset, and a 0 to indicate its absence. Another possibility (call it a_2) is to use 0 to indicate presence and 1 to indicate absence.

We are now in a position to say what we mean by an operator being invariant with respect to a collection of representations. We would like to say that the GA using such an operator will run the same, independent of the choice of representation. So, for example, uniform crossover works exactly the same whether we use a_1 or a_2 above. Given $i, j, k \in \Omega$, denote the probability that parents $\{i, j\}$ produce child k by $s(i, j, k)$.

[1]Readers interested in general methods for constructing crossover operators might also like to read Radcliffe (1992) and Moraglio et al. (2007).

Then, for crossover to be invariant with respect to representations a_1 and a_2, we require

$$s(a_1(x), a_1(y), a_1(z)) = s(a_2(x), a_2(y), a_2(z))$$

for all $x, y, z \in S$. Similarly, for mutation, write the probability that j mutates to i as $U(i, j)$. Then mutation is invariant with respect to a_1 and a_2 if

$$U(a_1(x), a_1(y)) = U(a_2(x), a_2(y))$$

for all $x, y \in S$.

Note that selection, which depends only on the objective value of an object, not its representation, is always invariant. In order to express this formally, consider that for any choice of representation $a : S \to \Omega$, we have a *fitness function* $f_a : \Omega \to \mathbb{R}$ defined on the search space, given by

$$f_a(i) = F(a^{-1}(i)).$$

So if we use representation a_1, then the fitness ascribed to an element representing $x \in S$ is $f_{a_1}(a_1(x)) = F(x)$, and is the same as the fitness we would get using representation a_2. The effects of any of the usual selection operators (proportional, tournament, rank) will therefore be the same, independent of the choice of representation.

We now give necessary and sufficient conditions for crossover and mutation to be invariant with respect to a set of representations, A. In order to do this, we consider functions of the form $a_1 \circ a_2^{-1}$, for $a_1, a_2 \in A$. Such functions are permutations of Ω. The collection of all such functions contains the identity function and is closed for taking inverses. We then collect all the functions that can be made by composing two or more of these functions together—these are again permutations of Ω. This collection forms a group $G(\Omega)$ acting on Ω, which we call the group action *generated* by A.

THEOREM 1: *Crossover is invariant with respect to a set of representations A if and only if*

$$s(i, j, k) = s(g(i), g(j), g(k))$$

for all $i, j, k \in \Omega$ and for all $g \in G(\Omega)$, the group action generated by A. In this case we say that crossover commutes *with the group action.*

A proof of the previous and following theorems (in a more general and abstract form) can be found in Rowe et al. (2002).

THEOREM 2: *Mutation is invariant with respect to a set of representations A if and only if*

$$U(i, j) = U(g(i), g(j))$$

for all $i, j \in \Omega$ and for all $g \in G(\Omega)$, the group action generated by A. In this case we say that mutation commutes *with the group action.*

The result of these theorems is that the search for genetic operators that are invariant with respect to a set of representations amounts to finding operators that commute with the group action generated by those representations. The main aim of this paper is to characterize operators that commute with a group action. We will, however, restrict

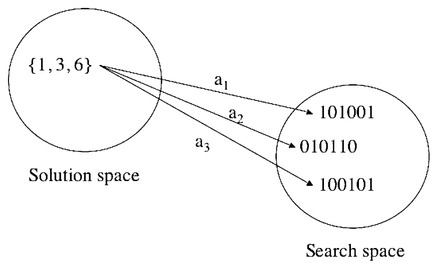

Figure 1: Three different representations of a subset as a binary string.

ourselves to considering *transitive* group actions. That is, given any two elements $i, j \in \Omega$, we will assume there is some group element $g \in G$ such that $i = g(j)$.

When a genetic operator commutes with a transitive group action, there are a number of consequences that follow. The interested reader is referred to Rowe et al. (2002, 2004, 2007) for details.

3 Examples

3.1 Subset Problems

When we represent subsets of some set using binary strings, we have already seen two types of invariance that may be relevant. First, there is invariance with respect to the choice of the label 0 or 1 to indicate the presence or absence of an element. Second, there is invariance with respect to the ordering of the elements of the set. Consider, as an example, the set $X = \{1, 2, 3, 4, 5, 6\}$. Our solution space S is the set of all subsets of X. We will be representing these subsets as binary strings of length six. The set of all such strings is the search space Ω. The most obvious representation is to use a 1 in each position to indicate the presence of an object, assuming numerical ordering. We will call this representation a_1. Figure 1 shows this representation for the subset $\{1, 3, 6\}$. Another representation, a_2, uses the same ordering but uses a 0 to indicate the presence of an object, rather than a 1. One of our group elements is $a_2 \circ a_1^{-1}$, which tells us how to translate strings from the first representation to the second. It is clear that this is done by taking the complement of the string. If for some bit positions we use 1 to indicate the presence of an element, and for others we use 0 for that purpose, then we can translate between two such representations by complementing just those bit positions where the two representations differ in their convention. The group elements arising from such relabellings therefore correspond to applying a bitwise exclusive-or with the appropriate binary mask. Given a binary mask b, the group element is:

$$g_b(i) = b \oplus i.$$

For the second type of invariance, consider the representation a_3 in Figure 1. Here we use the convention of always using a 1 to indicate the presence of an element, but the order of the elements of X have been reversed. Of course, other orderings are possible, and it is clear that translating between two such representations corresponds to permuting the bit positions of the strings accordingly. Such permutations give rise to another class of group element in our group action on Ω. The whole group is generated by these two classes of group element (permutations and exclusive-or masks).

We can now formally state the difference between the invariance properties of uniform and one-point crossover as follows. Uniform crossover commutes with the group generated by permutations and exclusive-or masks. One-point crossover does not commute with permutations, but does so with exclusive-or masks (which form a subgroup of the larger group action). This is a rather technical way of stating what we already said in the introduction, namely that using one-point crossover carries a commitment to a particular ordering, whereas uniform crossover is order independent. Both are invariant with respect to the choice of labels used at each bit position. Note that the larger the size of the group that the operator commutes with, the more invariance it has, and the fewer the representational commitments.

As an example, consider crossing the strings 100000 and 000001 using one-point crossover together with the permutation $x_1x_2x_3x_4x_5x_6 \mapsto x_2x_3x_4x_5x_6x_1$. Applying the permutation to the parents yields 000001 and 000010. Crossing, using the one-point crossover mask 111000, produces the offspring 000010. Reversing the permutation results in 000001, which could not possibly have been created from the original parents using one-point crossover alone.

3.2 Permutation Problems

Consider the traveling salesman problem (TSP), in which each element of the solution space is a tour of the set of cities. If there are n cities, then the search space Ω can be taken to be permutations of the numbers $\{1, 2, \ldots, n\}$. Different representations may correspond to different ways of numbering the cities, and translating between two representations would then correspond to appropriately permuting the numbering. If we wish our genetic operators to be invariant with respect to the numerical labeling of the cities, then we need them to commute with the group action given by those permutations; given such a permutation π, the corresponding group element $g : \Omega \to \Omega$ is

$$g_\pi(i) = \pi(i) = \pi \circ i$$

where $\circ$ is function composition and we interpret $i \in \Omega$ as itself a permutation of the set $\{1, 2, \ldots, n\}$.

Most crossover operators defined in the literature for TSP do have this property (Vose and Whitley, 1999). One that does not, however, is the *ordinal* crossover described in Michalewicz (1998). This explicitly makes use of an ordering of the cities in the problem. For example, suppose our TSP problem concerns the cities London, Oxford, Cambridge, and Birmingham, and we wish to cross over two parents representing the tours:

$$P1 = \text{Oxford, Birmingham, London, Cambridge}$$
$$P2 = \text{London, Oxford, Birmingham, Cambridge.}$$

In order to use ordinal crossover, we have to represent these tours numerically with respect to a reference ordering. Suppose we choose to represent tours with respect to the ordering:

$$R_1 = \text{London, Oxford, Cambridge, Birmingham}$$

then $P1 = (2, 3, 1, 1)$, meaning that we take the second element of R_1 first (and remove it from R_1). We then take the third element of the (remainder of) R_1. Then we take the first element of the remainder of that, and so on. Similarly $P2 = (1, 1, 2, 1)$. Crossing $P1$ and $P2$ with a one-point crossover in the middle produces offspring $(2, 3, 2, 1)$ which corresponds to the tour: Oxford, Birmingham, Cambridge, London.

If we chose a different representation ordering:

$$R_2 = \text{Birmingham, London, Cambridge, Oxford}$$

then $P1 = (4, 1, 1, 1)$ and $P2 = (2, 3, 1, 1)$. Applying exactly the same crossover operator produces offspring $(4, 1, 1, 1)$, which is the same as $P1$, namely, Oxford, Birmingham, London, Cambridge. We see that we get different results depending on which city ordering we choose, and so ordinal crossover is not invariant with respect to ordering.

3.3 Balanced Partition Problems

Our third example is a partitioning problem, in which a set of objects has to be divided into equal parts. Consider, for instance, the assignment of jobs evenly between machines, or dealing a pack of cards to a number of players. The problem is to find the optimal partition, as specified by a particular objective function. We will consider a simple example of six objects to be split three ways. The representations of solutions in the search space will be of the form $\langle\{u_1, u_2\}, \{v_1, v_2\}, \{w_1, w_2\}\rangle$, where each element in the list is a number from 1 to 6.

This example is instructive, as there are several different types of invariance that may be relevant depending on the particular problem at hand. One type of invariance we might hope for is with respect to the labeling of the objects so that it does not matter which job gets labeled $1, 2, \ldots$, the GA will run just the same. This invariance is given by group elements corresponding to permuting the labels; if π permutes the labels $\{1, 2, 3, 4, 5, 6\}$ then the corresponding group element is

$$g_\pi(\langle\{u_1, u_2\}, \{v_1, v_2\}, \{w_1, w_2\}\rangle) = \langle\{\pi(u_1), \pi(u_2)\}, \{\pi(v_1), \pi(v_2)\}, \{\pi(w_1), \pi(w_2)\}\rangle.$$

A second type of invariance is with respect to the labeling of the machines, which can be viewed as the ordering of the subsets of the partition. This is a similar issue to the ordering of bits in a binary string. Here the relevant group elements permute subsets, for example:

$$\langle\{u_1, u_2\}, \{v_1, v_2\}, \{w_1, w_2\}\rangle \longmapsto \langle\{v_1, v_2\}, \{w_1, w_2\}, \{u_1, u_2\}\rangle.$$

There are six possible ways to reorder the subsets. It is easily checked that any such reordering commutes with a permutation of the labels (it does not matter in which order they are performed, the result is the same).

A third type of invariance relates to the way we represent the subsets themselves. For example, if we use binary strings (of length six in this case) then we have the same issues as we did in the subset problem example. In order to simplify things, we will ignore this issue and assume our group is generated by the first two types of invariance. The group therefore contains $6 \times 6! = 4{,}320$ elements.

4 Characterizing Crossover and Mutation Operators

In this section we present the main results of the paper, namely the characterization of genetic operators that commute with a transitive group action. High-level algorithms for these operators are given, which we illustrate in the following section with our three examples. Proofs are postponed to following sections.

We assume a finite group G acts transitively on the search space Ω. We arbitrarily identify an element $\mathbf{o} \in \Omega$ as an *origin*. We then define the subgroup $\mathbf{Fix}(\mathbf{o})$ of G, known as the *stabilizer* of $\mathbf{o}$, as follows:

$$\mathbf{Fix}(\mathbf{o}) = \{g \in G | g(\mathbf{o}) = \mathbf{o}\}.$$

This subgroup is instrumental in characterizing crossover and mutation.

Crossover is implemented using functions from the search space to itself. Their role (as will be seen in the following section) is a generalization of the use of masks in the usual forms of crossover on binary strings. In particular, we will select such a function according to some probability distribution χ, which must satisfy

$$\chi(f) = \chi(d \circ f \circ d^{-1})$$

for all $f \in \Omega^{\Omega}$ and all $d \in \mathbf{Fix}(o)$. Given such χ, *symmetric* crossover—which is independent of the order of the parents—is given by the following algorithm. To crossover $\{i, j\} \subset \Omega$

1. With probability $1/2$, interchange i and j.
2. Let $v \in G$ be such that $v(\mathbf{o}) = j$.
3. Choose $f : \Omega \to \Omega$ according to χ.
4. Return $v(f(v^{-1}(i)))$.

A theorem in Section 6 shows this algorithm is independent of the particular choice of v in step 2, and the resulting crossover commutes with the group action. Moreover, any symmetric crossover that commutes with the group action can be implemented in this way (for some choice of distribution χ).

Mutation is implemented using elements of the search space. Their role (as will be seen in the following section) is a generalization of the use of masks in the usual forms of mutation on binary strings. In particular, we select such an element according to some probability distribution μ, which must satisfy

$$\mu(k) = \mu(d(k))$$

for all $k \in \Omega$ and all $d \in \mathbf{Fix}(\mathbf{o})$. Given such μ, mutation is implemented as follows. To mutate $j \in \Omega$

1. Let $v \in G$ be such that $v(\mathbf{o}) = j$.
2. Choose $c \in \Omega$ according to μ.
3. Return $v(c)$.

A theorem in Section 6 shows this algorithm is also independent of the particular choice of v in step 1 and that the resulting mutation commutes with the group action. Moreover, any mutation that commutes with the group action can be implemented in this way (for some choice of distribution μ).

5 Examples

5.1 Subset Problems

When representing subsets as binary strings, we first need to select an origin. The choice is arbitrary, but it is convenient to select the string of all zeros. The stabilizer **Fix**(**o**) contains all permutations of bit positions, but none of the exclusive-or elements (except, of course, exclusive-or with the origin itself, which is the identity). We now need to find, for each string $j \in \Omega$, some group element $g_j \in G$ such that $g_j(0) = j$. A simple choice is

$$g_j(x) = j \oplus x.$$

Note that for this choice $g_j^{-1} = g_j$ for all $j \in \Omega$.

In order to implement crossover, we need to choose a function from Ω to itself according to some preselected probability distribution χ that satisfies $\chi(f) = \chi(d \circ f \circ d^{-1})$ for all $d \in \mathbf{Fix}(\mathbf{o})$. Given string $b \in \Omega$, define the function $f_b : \Omega \to \Omega$ by

$$f_b(k) = b \otimes k$$

where $\otimes$ is a bitwise-and operation. The child of parents $i, j \in \Omega$ by way of f_b is

$$\begin{aligned}
g_j \circ f_b \circ g_j^{-1}(i) &= g_j \circ f_b(j \oplus i) \\
&= g_j(b \otimes (j \oplus i)) \\
&= g_j((b \otimes j) \oplus (b \otimes i)) \\
&= j \oplus (b \otimes j) \oplus (b \otimes i) \\
&= (\mathbf{1} \otimes j) \oplus (b \otimes j) \oplus (b \otimes i) \\
&= ((\mathbf{1} \oplus b) \otimes j) \oplus (b \otimes i) \\
&= (\overline{b} \otimes j) \oplus (b \otimes i)
\end{aligned}$$

where $\mathbf{1}$ is the string of all ones and $\overline{b}$ is the complement of b. Therefore, implementing crossover by choosing χ to give positive probability only to such functions f_b yields an instance of the usual mask-based crossover. Of course, many other kinds of crossover are also allowed by our theorem simply by varying the distribution χ.

Suppose we would like to use masks implementing one-point crossover. For strings of length 5, they are {00000, 10000, 11000, 11100, 11110, 11111}. The necessary condition on χ requires that if f_b is chosen with some probability (i.e., if crossover uses mask b), then we are obliged to choose with equal probability each of $d \circ f_b \circ d^{-1}$ for all $d \in \mathbf{Fix}(\mathbf{o})$. Taking $b = 11000$, for example, this means that masks 11000, 10100, 10010, 10001, 01100, 01010, 01001, 00110, 00101, and 00011 are all used with equal probability. In other words, to ensure invariance of crossover with respect to bit position ordering, we are forced to use some form of uniform crossover.

Note that if we did not include this form of invariance, then the group action would comprise just the exclusive-or group elements, and **Fix**(**o**) would contain only the identity. In that case we would be permitted to use masks such as 11100 without including all its permutations, and so one-point crossover would be permissible. In either case, crossover by masks is invariant with respect to the labeling convention of the individual bits. Remember that the smaller the group, the less invariance properties are ensured, and so the *greater* the representational commitment. This then allows us a corresponding degree of control in selecting which aspects of the representation we consider relevant (e.g., whether we have reason to believe that linkage between neighboring bit positions is a significant factor that ought to be respected by the crossover we use).

Implementing the usual bitwise mutation, with rate u, corresponds to choosing mutation masks $k \in \Omega$ with probability

$$\mu(k) = u^{\#k}(1-u)^{(\ell\text{-}\#k)}$$

(where $\#k$ is the number of ones in k). Since this only depends on the number of ones in the chosen mask, and not on their position, the condition $\mu(k) = \mu(d(k))$ is satisfied for all $d \in \mathbf{Fix}(\mathbf{o})$. This type of mutation is therefore invariant with respect to bit position ordering, as well as to the labeling convention for each bit.

Now consider the extreme situation of requiring mutation to be maximally invariant, that is, that it should commute with all permutations of the search space. **Fix**(**o**) is the set of all permutations that permute everything but **o**. The characterization theorem then tells us that the probability of mutating with any mask other than **o** must have uniform probability. That is, there is some fixed probability that a string is not mutated, otherwise it is mutated to any other string with equal probability. Searching with such a mutation operator therefore corresponds to performing random search.

5.2 Permutation Problems

Suppose we are representing tours by permutations of the numbers $\{1, 2, \ldots, n\}$, where n is the number of cities. Given a permutation π, we will write the corresponding tour as $\langle \pi(1), \pi(2), \ldots, \pi(n)\rangle$ so that $\pi(k)$ is the kth city in the tour. We arbitrarily select the identity permutation $\langle 1, 2, \ldots, n\rangle$ as a convenient origin; it enables us to easily find a group element such that $g_\pi(\mathbf{o}) = \pi$. Given tour π, define g_π by

$$g_\pi(\langle c_1, c_2, \ldots, c_n\rangle) = \langle \pi(c_1), \pi(c_2), \ldots, \pi(c_n)\rangle.$$

Because the group action corresponds to all reorderings of the cities, the ordering of the cities is completely arbitrary (unlike for ordinal crossover) and the GA will run *exactly* the same whichever ordering is used. Identifying **Fix**(**o**) is easy; it is the trivial subgroup containing just the identity. This means that we have no restrictions on our choice of probability distributions χ and μ. It also follows that the g_π defined above is the *only* possible choice of group element for π. We can use this fact to define a group operation $\oplus$ on the set Ω itself by setting

$$\pi_1 \oplus \pi_2 = g_{\pi_1}(\pi_2) = \pi_1 \circ \pi_2.$$

This situation, where the search space itself takes on the group structure, forms an important special case in the proof of our main theorem. Some of the consequences of this are described in Rowe et al. (2002, 2004, 2007).

In order to crossover two tours π_1 and π_2, we pick any $f \in \Omega^\Omega$ according to a predefined distribution χ and create the offspring

$$\pi_2 \circ f(\pi_2^{-1} \circ \pi_1).$$

Several crossovers defined for TSP in the literature are invariant with respect to the ordering of the cities (Vose and Whitley, 1999). It follows that they can be implemented as described above, by choosing χ appropriately.

We will illustrate the previous construction using *cyclic crossover* (CX), defined in Michalewicz (1998). Given two parents, CX is a deterministic method for producing an offspring in which each city's position comes from one of the parents. We start by taking the first city of the first parent, and copying this to the offspring. This means that we need to find a position for the first city of the second parent. There is only one choice, namely to place it according to the first parent. This sequence of choices continues until a cycle is complete. We then begin again with the next available city from the second parent. This continues, switching parents as each cycle is completed, until all the cities are placed. For example, if the parents are

$$P1 = \langle 4, 6, 5, 7, 2, 8, 1, 3, 9, 11, 12, 10\rangle$$
$$P2 = \langle 1, 10, 3, 4, 9, 11, 7, 5, 2, 6, 12, 8\rangle$$

then we start the offspring with 4, which forces the choice for 1, and then 7 all from the first parent. We then switch parents and place city 10 according to $P2$, which forces the place for 6, 11, and 8. We then switch back to $P1$ and continue. The resulting offspring is $\langle 4, 10, 5, 7, 9, 11, 1, 3, 2, 6, 12, 8\rangle$. In order to implement this crossover using our algorithm, we need to find the function $f : \Omega \to \Omega$ that will be used. Note that there will only be one function, as the procedure is deterministic. We can identify this function by noting that crossing a tour π with the identity gives the offspring $f(\pi)$; for CX, the effect of taking the second parent to be the identity makes it easy to determine the offspring. One writes out the first parent in cyclic notation, ordering the cycles by their smallest element, and including order-1 cycles. For example, $P1$ above can be written as (1 4 7)(2 6 8 3 5)(9)(10 11 12). We then select alternate cycles in this decomposition, starting with the first. That is, we get (1 4 7)(9), which corresponds to the tour $\langle 4\,2\,3\,7\,5\,6\,1\,8\,9\,10\,11\,12\rangle$. In order to verify this for the example above we calculate

$$\begin{aligned}
P2^{-1} &= \langle 1, 9, 3, 4, 8, 10, 7, 12, 5, 2, 6, 11\rangle\\
P2^{-1} \circ P1 &= \langle 4, 10, 8, 7, 9, 12, 1, 3, 5, 6, 11, 2\rangle\\
&= (1\,4\,7)(2\,10\,6\,12)(3\,8)(5\,9)(11)\\
f(P2^{-1} \circ P1) &= (1\,4\,7)(3\,8)(11)\\
&= \langle 4, 2, 8, 7, 5, 6, 1, 3, 9, 10, 11, 12\rangle\\
P2 \circ f(P2^{-1} \circ P1) &= \langle 4, 10, 5, 7, 9, 11, 1, 3, 2, 6, 12, 8\rangle
\end{aligned}$$

as previously constructed.

Implementing mutation using our general algorithm is straightforward, given our choice of origin (the only group element that maps the origin to itself is the identity). We can determine which "masks" $k \in \Omega$ should be used, and with what probability, by asking how the origin mutates, because $g_{\mathbf{o}}(k) = k$ (since $\mathbf{o}$ is the identity). For example, a common mutation operator is to reverse some section of the tour. The effect of reversing the first five cities of the origin $\langle 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 \rangle$ is simply $\langle 5, 4, 3, 2, 1, 6, 7, 8, 9, 10, 11, 12 \rangle$. This, then, is one of the elements of Ω that we will use as a mutation "mask." For example, mutating $P1$ above with this element gives

$$\begin{aligned} \langle 4, 6, 5, 7, 2, 8, 1, 3, 9, 11, 12, 10 \rangle &\circ \langle 5, 4, 3, 2, 1, 6, 7, 8, 9, 10, 11, 12 \rangle \\ &= \langle 2, 7, 5, 6, 4, 8, 1, 3, 9, 11, 12, 10 \rangle \end{aligned}$$

as required. The fact that tour segment reversal can be implemented in this way means that it is invariant with respect to the ordering of the cities.

5.3 Balanced Partition Problems

If we have $n = mk$ jobs to be assigned evenly to m different machines, then the number of ways to do this is $n!/(k!)^m$. Consider the example $n = 6$, $m = 3$, $k = 2$, so there are 90 possibilities. Note that we count $\langle \{u_1, u_2\}, \{v_1, v_2\}, \{w_1, w_2\} \rangle$ and $\langle \{v_1, v_2\}, \{u_1, u_2\}, \{w_1, w_2\} \rangle$ as different partitions since the jobs are being assigned to different machines. The search space can get big quickly (e.g., dividing 16 jobs equally between four different machines can be done in over 63 million ways).

Assuming the jobs are numbered 1 to 6, we will arbitrarily assign the origin to be $\langle \{1, 2\}, \{3, 4\}, \{5, 6\} \rangle$. Recall that the group action is generated by permutations of job labels and reorderings of machines. The label permutations that fix $\mathbf{o}$ form a subgroup generated by the label transpositions $(1\,2)$, $(3\,4)$, and $(5\,6)$. This subgroup, which we will denote D, has eight elements. Now for any machine reordering g, it is easy to find a label permutation π such that $g \circ \pi \in \mathbf{Fix}(\mathbf{o})$, and if $d \in D$ then $g \circ \pi \circ d \in \mathbf{Fix}(\mathbf{o})$. Suppose another permutation of labels π' has the property $g \circ \pi \in \mathbf{Fix}(\mathbf{o})$, so there are elements $a, a' \in \mathbf{Fix}(\mathbf{o})$ such that

$$a = g \circ \pi, \quad a' = g \circ \pi'.$$

Since $\mathbf{Fix}(\mathbf{o})$ is a subgroup, it follows that $a^{-1} \circ a' \in \mathbf{Fix}(\mathbf{o})$ and

$$a^{-1} \circ a' = \pi^{-1} \circ g^{-1} \circ g \circ \pi' = \pi^{-1} \circ \pi'.$$

Hence $\pi^{-1} \circ \pi'$ is some label permutation $d \in D$ which fixes $\mathbf{o}$. Therefore $\pi' = \pi \circ d$. It follows, then, that given any machine reordering g, there are a corresponding eight elements of $\mathbf{Fix}(\mathbf{o})$ of the form $g \circ \pi$, where π is a permutation of labels. Since there are six possible machine reorderings, it follows that $\mathbf{Fix}(\mathbf{o})$ contains 48 elements. This fact can be checked by appealing to the orbit-stabilizer theorem which, for transitive group actions, states that $|G| = |\Omega| \times |\mathbf{Fix}(\mathbf{o})|$.

In order to implement the generic crossover scheme for parents $i, j \in \Omega$, we first have to find a group element $g \in G$ such that $g(\mathbf{o}) = j$. In our example, there will be 48 such elements, since if g is one element, then so is $g \circ d$ for all $d \in \mathbf{Fix}(\mathbf{o})$. Second, we have to choose $f \in \Omega^{\Omega}$ according to some prespecified distribution χ which satisfies

$\chi(f) = \chi(d \circ f \circ d^{-1})$ for all $d \in \mathbf{Fix}(\mathbf{o})$. Thus for any one function we would like to use, there are 47 others we are obliged to use with the same probability.

6 Results

6.1 Definitions and Notation

Let Λ_Ω be the set of probability vectors over a finite search space Ω (each element is a column vector, indexed by Ω, whose components sum to one). Elements of Λ_Ω also represent populations; $p \in \Lambda_\Omega$ represents a population containing k in proportion p_k (we follow the usual convention that unbound variables are universally quantified). That is, p_k is the number of copies of item k found in the population divided by the population size.

A map $C : \Lambda_\Omega \to \Lambda_\Omega$ is a *quadratic operator* iff there exist symmetric matrices M_k such that

$$C(p)_k = p^T M_k p.$$

The quadratic operator C represents a binary crossover operator if $(M_k)_{i,j}$ is the probability $s(i, j, k)$ that child k is produced by parents $\{i, j\}$.

Let G be a finite group that acts transitively on Ω (we use multiplicative notation to represent the group operation). Choose an arbitrary element $\mathbf{o} \in \Omega$ to be the *origin*, except that if $\Omega = G$, then choose the origin to be the group identity element 1. For each $g \in G$, define the permutation matrix σ_g by

$$(\sigma_g)_{i,j} = [i = g(j)]$$

where [*expression*] denotes the integer 1 if *expression* is true, and otherwise denotes the integer 0. Crossover *commutes* with the group action (Rowe et al., 2002) iff

$$\sigma_g \circ C = C \circ \sigma_g.$$

In that case, the matrices M_k associated with C are determined by the *mixing matrix* $M = M_\mathbf{o}$, and if $g(\mathbf{o}) = k$ then

$$M_k = \sigma_g M \sigma_g^T.$$

Moreover, commutativity is equivalent to

$$s(g(i), g(j), g(k)) \;=\; s(i, j, k).$$

Similarly, a linear operator $U : \Lambda_\Omega \to \Lambda_\Omega$ represents a unary mutation operator if $U_{i,j}$ is the probability j mutates to i. Mutation commutes with the group action iff

$$\sigma_g \circ U = U \circ \sigma_g.$$

In the same way, crossover and mutation operators which commute with the action of G on itself (choose $\Omega = G$ in the explanation above) are defined with respect to Λ_G (the set of probability vectors over G). We are interested in projections of operators from

Λ_G to Λ_Ω. A function $f : \Lambda_G \to \Lambda_G$ is *compatible* with $\Xi : \Lambda_G \to \Lambda_\Omega$ iff there exists a function $\overline{f} : \Lambda_\Omega \to \Lambda_\Omega$ such that the following diagram commutes:

$$\begin{tikzcd}
\Lambda_G \arrow[r, "f"] \arrow[d, "\Xi"'] & \Lambda_G \arrow[d, "\Xi"'] \\
\Lambda_\Omega \arrow[r, "\bar{f}"] & \Lambda_\Omega
\end{tikzcd} .$$

6.2 Special Case: The Search Space Has a Group Structure

The case where $G = \Omega$ is a group acting on itself by translation ($g(i) = gi$) has already been sorted out in Rowe et al. (2007). The *canonical crossover scheme* is defined with respect to an arbitrary probability distribution χ (called the *crossover distribution*) over the set G^G of maps from G to itself. An offspring is produced from parents $i, j \in G$ as follows:

1. Choose an element $b \in G^G$ according to χ.
2. Return (with equal probability) an element from $\{jb(j^{-1}i),\ ib(i^{-1}j)\}$.

The main result of Rowe et al. (2007) is that the canonical crossover scheme commutes with G, and every two-parent crossover commuting with G is some instance of the canonical crossover scheme.

Without loss of generality, the crossover distribution χ satisfies $\chi_b = \chi_{\widehat{b}}$ (such distributions are called *symmetric*) where the map $\widehat{b} : G \to G$ is defined by

$$\widehat{b}(x) = xb(x^{-1}).$$

This follows from $b = \widehat{\widehat{b}}$ and the observation that the probability of obtaining k is

$$\begin{aligned}
&\frac{1}{2}\sum_b \chi_b[jb(j^{-1}i) = k] + \frac{1}{2}\sum_b \chi_b[ib(i^{-1}j) = k] \\
&= \frac{1}{2}\sum_b \chi_b[jb(j^{-1}i) = k] + \frac{1}{2}\sum_b \chi_b[i(i^{-1}j)\widehat{b}(j^{-1}i) = k] \\
&= \frac{1}{2}\sum_b \chi_b[jb(j^{-1}i) = k] + \frac{1}{2}\sum_{\widehat{b}} \chi_{\widehat{b}}[jb(j^{-1}i) = k] \\
&= \sum_b \frac{\chi_b + \chi_{\widehat{b}}}{2}[jb(j^{-1}i) = k].
\end{aligned}$$

If $\alpha_b = (\chi_b + \chi_{\widehat{b}})/2$, then $\alpha_b = \alpha_{\widehat{b}}$ and χ can be replaced with α in the chain of equalities above. Moreover, a consequence of assuming symmetric χ is that step 2 in the definition of the canonical crossover scheme can be replaced with:

2′. Return $jb(j^{-1}i)$.

6.3 Coarse Graining the Group to the Search Space

Let $C : \Lambda_G \to \Lambda_G$ be a crossover operator corresponding to an instance of the canonical crossover scheme; thus C commutes with G and has a mixing matrix M. Let the linear coarse graining operator $\Xi : \Lambda_G \to \Lambda_\Omega$ have matrix

$$\Xi_{i,g} = [i = g(\mathbf{o})].$$

Two elements $g, h \in G$ are said to be *equivalent* (denoted by $g \equiv h$) if $g(\mathbf{o}) = h(\mathbf{o})$. It follows that $h \equiv k \iff h = kd$ where $d \equiv 1$. Let e_k denote the kth column of the identity matrix. Then

$$g \equiv h \iff \Xi(e_g - e_h) = 0$$

(see Vose, 1999, for a basic discussion of equivalence in the context of linear coarse graining). The following theorem is in part concerned with the map $(d \cdot b) : G \to G$ defined by

$$(d \cdot b)(x) = b(xd).$$

Note that the map $d \mapsto d \cdot b$ induces a group action of G on G^G ($d \cdot (d' \cdot b) = (dd') \cdot b$ and $1 \cdot b = b$). In particular, the map $d \mapsto d \cdot b$ is invertible.

Because Ξ is linear, the compatibility of C reduces to the invariance of the kernel K of Ξ under the differential ∂C_x of C (see Rowe et al., 2006; for an alternate perspective on coarse graining, see Toussaint, 2005). If M is a matrix which is indexed by way of a group, its *twist* M^* is defined by

$$(M^*)_{i,j} \;=\; M_{i^{-1}j,i^{-1}}.$$

The derivation of ∂C_x given in Vose (1999) carries over (with trivial modifications) to the present context and yields

$$\partial C_x \;=\; 2\sum_g \sigma_g M^* \sigma_g^T\, x_g. \tag{1}$$

Therefore, K is invariant under ∂C_x provided it is invariant under σ_g and M^*. An element $v \in K$ is characterized by

$$0 \;=\; (\Xi v)_i \;=\; \sum_{g'} [i = g'(\mathbf{o})]\, v_{g'}.$$

It follows that K is invariant under σ_g, since $\Xi\sigma_g v$ has ith component

$$\begin{aligned}\sum_{g'}[i = g'(\mathbf{o})]\sum_{g''}[g' = gg'']v_{g''} &= \sum_{g'}[i = g'(\mathbf{o})]v_{g^{-1}g'}\\ &= \sum_{g'}[i = gg'(\mathbf{o})]v_{g'}\\ &= \sum_{g'}[g^{-1}(i) = g'(\mathbf{o})]v_{g'}.\end{aligned}$$

The considerations above lead to the following.

THEOREM 3: *A canonical crossover is compatible with Ξ iff the kernel of Ξ is invariant under the twist of the mixing matrix. Moreover, a sufficient condition for compatibility is that the crossover distribution χ satisfy $d \equiv 1 \Longrightarrow \chi_b = \chi_{d \cdot b}$.*

PROOF: We have just shown that invariance of the kernel of Ξ under the twist of the mixing matrix is a sufficient condition for crossover to be compatible with Ξ. Conversely, suppose canonical crossover C is compatible with Ξ, and therefore the kernel of Ξ is invariant under the differential ∂C_x at any point x. Let $z \in \Lambda_G$ be the vector given by $z_g = [g = 1]$, where 1 is the group identity element. From Equation (1), we see that $\partial C_z = 2M^*$. It follows that the kernel of Ξ is invariant under the twist M^*.

Now assume that $\chi_b = \chi_{d \cdot b}$, let $v \in K$, and let R be a complete set of equivalence class representatives (for the equivalence relation $\equiv$). Suppose that

$$h \equiv k \Longrightarrow s(h, 1, g) = s(k, 1, g).$$

The ith component of $\Xi M^* v$ is then

$$\begin{aligned}
\sum_g [i = g(\mathbf{o})](M^* v)_g &= \sum_g [i = g(\mathbf{o})] \sum_{g'} M_{g^{-1}g', g^{-1}} v_{g'} \\
&= \sum_g [i = g(\mathbf{o})] \sum_{g'} s(g^{-1}g', g^{-1}, 1) v_{g'} \\
&= \sum_g [i = g(\mathbf{o})] \sum_{g'} s(g', 1, g) v_{g'} \\
&= \sum_g [i = g(\mathbf{o})] \sum_{r \in R} \sum_{g' \equiv r} s(g', 1, g) v_{g'} \\
&= \sum_g [i = g(\mathbf{o})] \sum_{r \in R} \sum_{g' \equiv r} s(r, 1, g) v_{g'} \\
&= \sum_g [i = g(\mathbf{o})] \sum_{r \in R} s(r, 1, g) \sum_{g' \equiv r} v_{g'}.
\end{aligned}$$

Thus K is invariant under M^* since the inner sum above is zero, which can be seen as follows

$$\begin{aligned}
0 &= \sum_{g'} [r(\mathbf{o}) = g'(\mathbf{o})] v_{g'} \\
&= \sum_{r' \in R} \sum_{g' \equiv r'} [r(\mathbf{o}) = g'(\mathbf{o})] v_{g'} \\
&= \sum_{r' \in R} [r(\mathbf{o}) = r'(\mathbf{o})] \sum_{g' \equiv r'} v_{g'} \\
&= \sum_{r' \in R} [r \equiv r'] \sum_{g' \equiv r'} v_{g'} \\
&= \sum_{g' \equiv r} v_{g'}.
\end{aligned}$$

By what has been established, it only remains to show $h \equiv k \implies s(h,1,g) = s(k,1,g)$. Recall that $h \equiv k \implies h = kd$ where $d \equiv 1$. Hence (by the definition of canonical crossover and the assumption $h \equiv k$)

$$\begin{aligned} s(h,1,g) &= \sum_b \chi_b[b(h) = g] \\ &= \sum_b \chi_b[d \cdot b(k) = g] \\ &= \sum_b \chi_{d^{-1}\cdot b}[d \cdot (d^{-1} \cdot b)(k) = g] \\ &= \sum_b \chi_{d^{-1}\cdot b}[b(k) = g] \\ &= s(k,1,g) \end{aligned}$$

since $k^{-1}h = d \equiv 1 \implies d^{-1} \equiv 1$ so that $\chi_{d^{-1}\cdot b} = \chi_b$. □

LEMMA 4: *If a canonical crossover is compatible with Ξ, then*

$$x \equiv x' \implies \sum_g [i = g(\mathbf{o})] s(x,y,g) = \sum_g [i = g(\mathbf{o})] s(x',y,g).$$

PROOF: Let e_k denote the kth column of the identity matrix. It follows from Equation (1) that

$$\begin{aligned} \tfrac{1}{2}(\partial C_{e_y})_{i,j} &= \left(\sigma_y^T e_i\right)^T M^* \left(\sigma_y^T e_j\right) \\ &= e_{y^{-1}i}^T M^* e_{y^{-1}j} \\ &= M^*_{y^{-1}i,y^{-1}j} \\ &= M_{i^{-1}j,i^{-1}y} \\ &= s(i^{-1}j, i^{-1}y, 1) \\ &= s(j,y,i). \end{aligned}$$

If $x \equiv x'$ then $e_x - e_{x'} \in K$ and so compatibility implies $\partial C_{e_y}(e_x - e_{x'}) \in K$. Therefore $\Xi \partial C_{e_y} e_x$ is invariant under $x \mapsto x'$. But according to what has been established above,

$$\begin{aligned} 2(\Xi \partial C_{e_y} e_x)_i &= 2 \sum_g \Xi_{i,g} (\partial C_{e_y})_{g,x} \\ &= \sum_g [i = g(\mathbf{o})] s(x,y,g). \end{aligned}$$

□

A canonical crossover C satisfying the conditions of Theorem 3 is compatible with Ξ and commutes with G. The linear maps σ_g are also compatible with Ξ since they preserve K (this was shown before Theorem 3). Therefore, there exist coarse grainings

$\overline{C}$ and $\overline{\sigma_g}$ such that $\overline{C} \circ \Xi = \Xi \circ C$ and $\overline{\sigma_g} \circ \Xi = \Xi \circ \sigma_g$. It follows that

$$\overline{C} \circ \overline{\sigma_g} \circ \Xi = \overline{C} \circ \Xi \circ \sigma_g = \Xi \circ C \circ \sigma_g = \Xi \circ \sigma_g \circ C = \overline{\sigma_g} \circ \Xi \circ C = \overline{\sigma_g} \circ \overline{C} \circ \Xi$$

and therefore

$$\overline{C} \circ \overline{\sigma_g} = \overline{\sigma_g} \circ \overline{C}.$$

The coarse grainings are given explicitly (see Vose, 1999) by

$$\begin{aligned} \overline{C} &= \Xi \circ C \circ D \circ \Xi^T \\ \overline{\sigma_g} &= \Xi \circ \sigma_g \circ D \circ \Xi^T \end{aligned}$$

where in the present context $D = \alpha^{-1} I$, and $\alpha = |\{d \; : \; d \equiv 1\}|$ is the size of an equivalence class. In particular,

$$\begin{aligned} (\overline{\sigma_g})_{i,j} &= (\Xi \sigma_g D \Xi^T)_{i,j} \\ &= \alpha^{-1} \sum_{h,k} [i = h(\mathbf{o})]\,[h = gk]\,[j = k(\mathbf{o})] \\ &= \alpha^{-1} \sum_{k} [i = gk(\mathbf{o})]\,[j = k(\mathbf{o})] \\ &= [i = g(j)]\,\alpha^{-1} \sum_{k} [j = k(\mathbf{o})] \\ &= [i = g(j)]. \end{aligned}$$

Thus $\overline{\sigma_g}$ is the matrix of the group action on Λ_Ω, and therefore $\overline{C}$ commutes with G. The considerations above lead to the following.

THEOREM 5: *If a canonical crossover C with mixing matrix M is compatible with Ξ, then its quotient $\overline{C}$ commutes with G and has mixing matrix $\overline{M}$ defined by*

$$\overline{s}(i, j, \mathbf{o}) \; = \; \overline{M}_{i,j} \; = \; \sum_{g \equiv 1} M_{g^{-1}r, g^{-1}r'}$$

where $\overline{s}(i, j, \mathbf{o})$ is the probability parents $\{i, j\}$ produce child $\mathbf{o}$, and where r, r' are arbitrary elements of G such that $r(\mathbf{o}) = i, \; r'(\mathbf{o}) = j$.

PROOF: Except for the claim concerning $\overline{M}$, the theorem has already been established. If R is a complete set of equivalence class representatives (for the equivalence relation $\equiv$), then

$$\begin{aligned} \overline{C}(x)_k &= (\Xi C(D \Xi^T x))_k \\ &= \sum_g [k = g(\mathbf{o})] C(D \Xi^T x)_g \\ &= \alpha^{-2} \sum_g [k = g(\mathbf{o})] (\Xi^T x)^T \sigma_g M \sigma_g^T (\Xi^T x) \end{aligned}$$

$$
\begin{aligned}
&= \alpha^{-2} \sum_g [k = g(\mathbf{o})] \sum_a \sum_b \left(\sigma_g^T \, \Xi^T x\right)_a \left(\sigma_g^T \, \Xi^T x\right)_b M_{a,b} \\
&= \alpha^{-2} \sum_g [k = g(\mathbf{o})] \sum_a \sum_b (\Xi^T x)_{ga} (\Xi^T x)_{gb} \, M_{a,b} \\
&= \alpha^{-2} \sum_g [k = g(\mathbf{o})] \sum_a \sum_b x_{ga(\mathbf{o})} x_{gb(\mathbf{o})} \, M_{a,b} \\
&= \alpha^{-2} \sum_g [k = g(\mathbf{o})] \sum_a \sum_b x_{a(\mathbf{o})} x_{b(\mathbf{o})} \, M_{g^{-1}a, g^{-1}b} \\
&= \alpha^{-2} \sum_g [k = g(\mathbf{o})] \sum_{r \in R} \sum_{r' \in R} \sum_{d \equiv 1} \sum_{d' \equiv 1} x_{rd(\mathbf{o})} x_{r'd'(\mathbf{o})} \, M_{g^{-1}rd, g^{-1}r'd'} \\
&= \alpha^{-2} \sum_{r \in R} \sum_{r' \in R} x_{r(\mathbf{o})} x_{r'(\mathbf{o})} \sum_{d \equiv 1} \sum_{d' \equiv 1} \sum_g [k = g(\mathbf{o})] s(g^{-1}rd, g^{-1}r'd', 1) \\
&= \alpha^{-2} \sum_{r \in R} \sum_{r' \in R} x_{r(\mathbf{o})} x_{r'(\mathbf{o})} \sum_{d \equiv 1} \sum_{d' \equiv 1} \sum_g [k = g(\mathbf{o})] s(rd, r'd', g).
\end{aligned}
$$

Since $rd \equiv r$ and $r'd' \equiv r'$ (because $d \equiv d' \equiv 1$), appealing to lemma 4 and the fact that $s(x, y, z) = s(y, x, z)$, shows the inner sum above is

$$\sum_g [k = g(\mathbf{o})] s(r, r', g).$$

It follows that

$$
\begin{aligned}
\overline{C}(x)_{\mathbf{o}} &= \sum_{r \in R} \sum_{r' \in R} x_{r(\mathbf{o})} x_{r'(\mathbf{o})} \sum_g [\mathbf{o} = g(\mathbf{o})] s(r, r', g) \\
&= \sum_{r \in R} \sum_{r' \in R} x_{r(\mathbf{o})} x_{r'(\mathbf{o})} \sum_{g \equiv 1} M_{g^{-1}r, g^{-1}r'}.
\end{aligned}
$$

□

THEOREM 6: *Let a canonical crossover with symmetric distribution χ be compatible with Ξ. The quotient crossover can be implemented as follows. To produce an offspring from parents $\{i, j\} \subset \Omega$,*

1. *Let $r_i, r_j \in G$ be such that $i = r_i(\mathbf{o})$, $j = r_j(\mathbf{o})$.*
2. *Pick $b \in G^G$ according to χ.*
3. *Return $r_j b(r_j^{-1} r_i)(\mathbf{o})$.*

PROOF: Let $r_k \in G$ be such that $r_k(\mathbf{o}) = k$. The probability that k is produced is

$$
\begin{aligned}
\sum_b \chi_b \left[r_j b\left(r_j^{-1} r_i\right)(\mathbf{o}) = r_k(\mathbf{o})\right] &= \sum_b \chi_b \left[r_j b\left(r_j^{-1} r_i\right) \equiv r_k\right] \\
&= \sum_{g \equiv 1} \sum_b \chi_b \left[r_j b\left(r_j^{-1} r_i\right) = r_k g\right]
\end{aligned}
$$

$$
\begin{aligned}
&= \sum_{g \equiv 1} s(r_i, r_j, r_k g) \\
&= \sum_{g \equiv 1} s(g^{-1} r_k^{-1} r_i,\ g^{-1} r_k^{-1} r_j, 1) \\
&= \sum_{g \equiv 1} M_{g^{-1} r_k^{-1} r_i,\, g^{-1} r_k^{-1} r_j}.
\end{aligned}
$$

Since $r_k^{-1} r_i(\mathbf{o}) = r_k^{-1}(i)$ and $r_k^{-1} r_j(\mathbf{o}) = r_k^{-1}(j)$, appealing to Theorem 5 shows the last expression above is

$$\overline{M}_{r_k^{-1}(i),\, r_k^{-1}(j)} \;=\; \overline{s}\big(r_k^{-1}(i), r_k^{-1}(j), \mathbf{o}\big) \;=\; \overline{s}(i, j, k).$$

□

THEOREM 7: *Every crossover operator on Λ_Ω commuting with G is a quotient $\overline{C}$ of a canonical crossover C on Λ_G with symmetric crossover distribution χ satisfying $d \equiv 1 \Longrightarrow \chi_b = \chi_{d \cdot b}$.*

PROOF: Let $\overline{s}(i, j, k)$ denote the probability that parents $\{i, j\}$ produce child k (for $i, j, k \in \Omega$), and for $t, u, v \in G$ define the probability $s(t, u, v)$ that parents $\{t, u\}$ produce child v by

$$s(t, u, v) \;=\; \overline{s}(t(\mathbf{o}), u(\mathbf{o}), v(\mathbf{o}))/\alpha.$$

By hypothesis, the crossover corresponding to $\overline{s}$ commutes with G; therefore

$$s(gt, gu, gv) \;=\; \overline{s}(gt(\mathbf{o}), gu(\mathbf{o}), gv(\mathbf{o}))/\alpha \;=\; \overline{s}(t(\mathbf{o}), u(\mathbf{o}), v(\mathbf{o}))/\alpha \;=\; s(t, u, v).$$

Hence there exists a canonical crossover C with mixing matrix M corresponding to s. According to Theorem 5, the quotient $\overline{C}$ is the crossover operator we began with, since

$$
\begin{aligned}
\overline{M}_{i,j} &= \sum_{g \equiv 1} M_{g^{-1} r, g^{-1} r'} \quad \text{where } r(\mathbf{o}) = i,\ r'(\mathbf{o}) = j \\
&= \sum_{g \equiv 1} s(r, r', g) \\
&= \alpha^{-1} \sum_{g \equiv 1} \overline{s}(r(\mathbf{o}), r'(\mathbf{o}), g(\mathbf{o})) \\
&= \overline{s}(i, j, \mathbf{o}).
\end{aligned}
$$

The properties of χ remain to be verified (according to Theorem 3, $d \equiv 1 \Longrightarrow \chi_b = \chi_{d \cdot b}$ implies C is compatible with Ξ). The proofs of Theorems 5 and 6 of Rowe et al. (2007) show the crossover distribution can be defined as $\chi_b = (\alpha_b + \alpha_{\widehat{b}})/2$ where

$$\alpha_b \;=\; \prod_g M_{b(g)^{-1},\, b(g)^{-1} g} \;=\; \prod_g s(1, g, b(g)) \;=\; \prod_g \overline{s}(\mathbf{o}, g(\mathbf{o}), b(g)(\mathbf{o}))/\alpha. \tag{2}$$

Making use of Equation (2) yields

$$\begin{aligned}\alpha_{\widehat{b}} &= \prod_g \bar{s}(\mathbf{o}, g(\mathbf{o}), gb(g^{-1})(\mathbf{o}))/\alpha \\ &= \prod_g \bar{s}(g^{-1}(\mathbf{o}), \mathbf{o}, b(g^{-1})(\mathbf{o}))/\alpha \\ &= \prod_g \bar{s}(g(\mathbf{o}), \mathbf{o}, b(g)(\mathbf{o}))/\alpha \\ &= \alpha_b.\end{aligned}$$

In particular, $\chi = \alpha$. If $d \equiv 1$ (and therefore $d^{-1}(\mathbf{o}) = \mathbf{o}$), then Equation (2) gives

$$\begin{aligned}\alpha_{d\cdot b} &= \prod_g \bar{s}(\mathbf{o}, g(\mathbf{o}), b(gd)(\mathbf{o}))/\alpha \\ &= \prod_g \bar{s}(\mathbf{o}, gd^{-1}(\mathbf{o}), b(gd^{-1}d)(\mathbf{o}))/\alpha \\ &= \prod_g \bar{s}(\mathbf{o}, g(\mathbf{o}), b(g)(\mathbf{o}))/\alpha \\ &= \alpha_b.\end{aligned}$$

□

6.4 Using Maps on the Search Space

Taken together, Theorems 3, 5, 6, and 7 have the dual function of characterizing and providing an implementation for those crossovers on Λ_Ω which commute with G. This section will conclude with an alternate characterization and implementation involving functions from Ω^Ω rather than from G^G.

DEFINITION 2: *Define R_k for $k \in \Omega$ by*

$$R_k = \{g \in G \mid g(\mathbf{o}) = k\}.$$

Define $T(k, b, f)$ for $k \in \Omega$, $b : G \to G$, and $f : \Omega \to \Omega$ by

$$T(k, b, f) = \{g \in R_k \mid b(g)(\mathbf{o}) = f(k)\}.$$

Given crossover distribution χ on G^G, define the corresponding function $\overline{\chi}$ on Ω^Ω by

$$\overline{\chi}(f) = \sum_b \chi_b \prod_k |T(k, b, f)|/\alpha.$$

LEMMA 8: *The function $\overline{\chi}$ is a probability distribution on Ω^Ω. If χ is symmetric and satisfies $d \equiv 1 \Longrightarrow \chi_b = \chi_{d\cdot b}$, then $\overline{\chi}$ satisfies*

$$d \equiv 1 \Longrightarrow \overline{\chi}(f) = \overline{\chi}(d \circ f \circ d^{-1}).$$

PROOF: Note that $D_{\mathbf{o}} = \{d : d \equiv 1\}$, and all R_k have size α. Therefore,[2]

$$\begin{aligned}
\sum_f \prod_k |T(k, b, f)|/\alpha &= \sum_{f \in \Omega^\Omega} \prod_{k \in \Omega} \sum_{g_k \in R_k} [b(g_k)(\mathbf{o}) = f(k)]/\alpha \\
&= \prod_{k \in \Omega} \sum_{f_k \in \Omega} \sum_{g_k \in R_k} [b(g_k)(\mathbf{o}) = f_k]/\alpha \\
&= \prod_{k \in \Omega} \sum_{g_k \in R_k} \sum_{f_k \in \Omega} [b(g_k)(\mathbf{o}) = f_k]/\alpha \\
&= \prod_{k \in \Omega} \sum_{g_k \in R_k} \alpha^{-1} \\
&= 1.
\end{aligned}$$

It follows that

$$\sum_f \overline{\chi}(f) = \sum_b \chi_b \sum_f \prod_k |T(k, b, f)|/\alpha = \sum_b \chi_b = 1.$$

Thus $\overline{\chi}$ is a probability distribution. Note that

$$\widehat{d \cdot \widehat{b}}(x) = x(d \cdot \widehat{b})(x^{-1}) = x\widehat{b}(x^{-1}d) = xx^{-1}db(d^{-1}x) = d \circ b \circ d^{-1}(x).$$

Hence a consequence of the assumptions regarding χ is that if $d \equiv 1$ and $b = d \circ b' \circ d^{-1}$, then $\chi_b = \chi_{b'}$. Therefore,

$$\begin{aligned}
\overline{\chi}(d \circ f \circ d^{-1}) &= \sum_b \chi_b \prod_k \sum_{g \in R_k} [b(g)(\mathbf{o}) = d \circ f \circ d^{-1}(k)]/\alpha \\
&= \sum_b \chi_b \prod_k \sum_{g \in R_{d(k)}} [b(g)(\mathbf{o}) = d \circ f(k)]/\alpha \\
&= \sum_b \chi_b \prod_k \sum_{h \in R_k} [(d^{-1} \circ b \circ d)(h)(\mathbf{o}) = f(k)]/\alpha \\
&= \sum_{b'} \chi_{b'} \prod_k \sum_{h \in R_k} [b'(h)(\mathbf{o}) = f(k)]/\alpha \\
&= \sum_{b'} \chi_{b'} \prod_k \sum_{h \in R_k} [b'(h)(\mathbf{o}) = f(k)]/\alpha \\
&= \overline{\chi}(f).
\end{aligned}$$

□

THEOREM 9: *Let π be a probability distribution on Ω^Ω such that $d \equiv 1 \Rightarrow \pi(f) = \pi(d \circ f \circ d^{-1})$, and implement crossover as follows. In order to produce an offspring from parents $\{i, j\} \subset \Omega$,*

[2]See appendix for details of summing over the set of functions.

1. *With probability 1/2, interchange i and j.*
2. *Let $v \in G$ be such that $v(\mathbf{o}) = j$.*
3. *Choose $f : \Omega \to \Omega$ according to π.*
4. *Return $v(f(v^{-1}(i)))$.*

The induced crossover operator on Λ_Ω commutes with G.

PROOF: We first show the crossover operator is well-defined (i.e., it is independent of the choice of v in step 2). Let $r(i, j, k)$ denote the probability that parents i, j produce child k using steps 2 through 4 above (i.e., parents i and j are *not* interchanged). It follows that

$$s(i, j, k) = \frac{r(i, j, k) + r(j, i, k)}{2}.$$

Moreover, $s(i, j, k)$ is independent of the choice of v in step 2 provided $r(i, j, k)$ is. Note that if v denotes the chosen representative for j in step 2, then

$$r(i, j, k) = \sum_f \pi(f)[v(f(v^{-1}(i))) = k].$$

Alternate representatives for j in step 2 have the form vd for some $d \equiv 1$, and lead to the production of offspring k with probability

$$\begin{aligned}\sum_f \pi(f)[(vd)(f((vd)^{-1}(i))) = k] &= \sum_f \pi(f)[v \circ d \circ f \circ d^{-1} \circ v^{-1}(i) = k]\\ &= \sum_f \pi(d^{-1} \circ f \circ d)[v \circ f \circ v^{-1}(i) = k]\\ &= \sum_f \pi(f)[v(f(v^{-1}(i))) = k].\end{aligned}$$

Crossover commutes with G will now be shown. If $v(\mathbf{o}) = g(j)$, then

$$\begin{aligned}r(g(i), g(j), g(k)) &= \sum_f \pi(f)[v(f(v^{-1}(g(i)))) = g(k)]\\ &= \sum_f \pi(f)[(g^{-1}v)f((g^{-1}v)^{-1}(i)) = k]\\ &= r(i, j, k).\end{aligned}$$

The last equality above is justified by the observation that $(g^{-1}v)(\mathbf{o}) = j$. □

THEOREM 10: *Every crossover operator on Λ_Ω commuting with G is an instance of the crossover defined in Theorem 9.*

PROOF: According to Theorem 6, Theorem 7, and Lemma 8, it suffices to show that the crossover operator described by Theorem 6 is the same as the crossover operator described by Theorem 9 for the choice $\pi = \overline{\chi}$. Reusing notation from the proof of Theorem 9,

$$\begin{aligned}
r(i,j,k) &= \sum_f \overline{\chi}(f)[v(f(v^{-1}(i))) = k] \\
&= \sum_f [v(f(v^{-1}(i))) = k] \sum_b \chi_b \prod_m |T(m,b,f)|/\alpha \\
&= \sum_b \chi_b \sum_f [v(f(v^{-1}(i))) = k] \prod_m \sum_{g_m \in R_m} [b(g_m)(\mathbf{o}) = f(m)]/\alpha \\
&= \sum_b \chi_b \prod_m \sum_{f_m} \left[v\left(f_{v^{-1}(i)}\right) = k\right] \sum_{g_m \in R_m} [b(g_m)(\mathbf{o}) = f_m]/\alpha \\
&= \sum_b \chi_b \sum_{f_{v^{-1}(i)}} \left[v\left(f_{v^{-1}(i)}\right) = k\right] \sum_{g \in R_{v^{-1}(i)}} \left[b(g)(\mathbf{o}) = f_{v^{-1}(i)}\right]/\alpha \\
&= \sum_b \chi_b \sum_g [g(\mathbf{o}) = v^{-1}(i)] \sum_{a \in \Omega} [b(g)(\mathbf{o}) = a][v(a) = k]/\alpha \\
&= \sum_b \chi_b \sum_g [g(\mathbf{o}) = v^{-1}(i)][v(b(g)(\mathbf{o})) = k]/\alpha \\
&= \sum_b \chi_b \sum_g [g(\mathbf{o}) = i][v(b(v^{-1}g)(\mathbf{o})) = k]/\alpha \\
&= \sum_g [g(\mathbf{o}) = i]/\alpha \sum_b \chi_b [v(b(v^{-1}g)(\mathbf{o})) = k].
\end{aligned}$$

The inner sum is the probability that crossing i and j produces k using the crossover described by Theorem 6, because $v(\mathbf{o}) = j$ and $g(\mathbf{o}) = i$. The outer sum over g is not relevant; there are α nonzero terms which correspond to alternate representatives g for i (and the choice of representative is irrelevant). □

6.5 Corollary: Mutation

Any linear operator $U : \Lambda_\Omega \to \Lambda_\Omega$ is a quadratic operator with associated matrices

$$s(i,j,k) \;=\; (M_k)_{i,j} \;=\; (U_{k,i} + U_{k,j})/2$$

(see Rowe et al., 2002). Therefore, mutation is a degenerate form of crossover. Mutation commutes with G iff

$$U_{g(i),g(j)} \;=\; U_{i,j}.$$

In that case, $s(g(i), g(j), g(k)) = s(i,j,k)$. Thus, any mutation operator commuting with G is implementable as crossover which commutes with G. That implies it suffices to construct an instance of the crossover defined in Theorem 9 such that

$$s(\mathbf{o}, \mathbf{o}, v^{-1}(k)) \;=\; U_{v^{-1}(k),\mathbf{o}}$$

since if $j = v(\mathbf{o})$, then

$$s(j, j, k) \;=\; U_{k,j}$$

and the result k of mutating j is obtained as the offspring of parents $\{j, j\}$.

For each $c \in \Omega$, define the constant function

$$f_c(\cdot) = c$$

and let π give positive probability to only those functions. The probability parents $\{\mathbf{o}, \mathbf{o}\}$ produce child $v^{-1}(k)$ is

$$\begin{aligned} s(\mathbf{o}, \mathbf{o}, v^{-1}(k)) &= \sum_f \pi(f)[1(f(1^{-1}(\mathbf{o}))) = v^{-1}(k)] \\ &= \sum_c \pi(f_c)[c = v^{-1}(k)] \\ &= \pi(f_{v^{-1}(k)}). \end{aligned}$$

Let μ be the probability distribution on Ω defined by $\mu(c) = \pi(f_c) = U_{c,\mathbf{o}}$. Then

$$s(\mathbf{o}, \mathbf{o}, v^{-1}(k)) \;=\; \pi(f_{v^{-1}(k)}) \;=\; U_{v^{-1}(k),\mathbf{o}}$$

as required. Moreover, the condition

$$d \equiv 1 \Longrightarrow \pi(f_c) = \pi(d \circ f_c \circ d^{-1})$$

is equivalent to

$$d \equiv 1 \Longrightarrow \mu(c) = \mu(d(c))$$

which is satisfied because $U_{i,j} = U_{g(i),g(j)}$ (choose $i = c$, $j = \mathbf{o}$, $g = d$).

As noted above, the result k of mutating j is obtained as the offspring of parents $\{j, j\}$. In this case the algorithm in Theorem 9 reduces to:

1. Let $v \in G$ be such that $v(\mathbf{o}) = j$.
2. Choose $c \in \Omega$ according to μ.
3. Return $v(c)$.

7 Conclusions

Crossover and mutation operators that commute with a transitive group action on the search space have been characterized. Moreover, it has been shown how to implement them in a representation-independent manner. If the group action is generated by a set of representations, then these operators are guaranteed to be invariant with respect to changes within the set of representations. These results have been illustrated with a number of examples.

An interesting feature of the proof is that the crossover and mutation operators arise as coarse grainings of genetic operators defined on the group itself. This could be taken a step further by considering projections of selection operators. If fitness is assigned to group elements such that $g \in G$ has the fitness of $g(\mathbf{o})$ in Ω, then equivalent group elements have equal fitness. Selection methods, such as proportional, binary tournament, and rank-based selection, will then project (in a form-invariant way) from the group level to the search space (see Rowe et al., 2006). This means any genetic algorithm (comprising selection, crossover, and mutation) running on the search space (where crossover and mutation commute with a transitive group action) is a coarse graining of a genetic algorithm running on the group itself (again with the crossover and mutation operators commuting with the group).

Acknowledgments

The authors are grateful to the participants of the Dagstuhl Seminar "Theory of Evolutionary Algorithms" (seminar 08051, January 2008) for helpful discussions on this topic. Vose was partially supported by National Institutes of Health grant "New approaches to the modeling of speciation" (GM56693). Vose and Rowe were funded under the NSF EPSCOR scheme to visit the University of Montana. The University of Birmingham has generously supported visits from Vose and Wright.

References

Michalewicz, Z. (1998). *Genetic algorithms + data structures = evolution programs*. Berlin: Springer.

Moraglio, A., Kim, H.-Y., Yoon, Y., and Moon, B.-R. (2007). Geometric crossovers for multiway graph partitioning. *Evolutionary Computation*, 14(4):445–474.

Radcliffe, N. J. (1992). The algebra of genetic algorithms. *Annals of Mathematics and Artificial Intelligence*, 10:339–384.

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

Rowe, J. E., Vose, M. D., and Wright, A. H. (2004). Structural search spaces and genetic operators. *Evolutionary Computation*, 12:461–494.

Rowe, J. E., Vose, M. D., and Wright, A. H. (2006). Differentiable coarse graining. *Theoretical Computer Science*, 361:111–129.

Rowe, J. E., Vose, M. D., and Wright, A. H. (2007). Neighbourhood graphs and symmetric genetic operators. In *Foundations of Genetic Algorithms 9, LNCS 4436*, pp. 110–122.

Toussaint, M. (2005). *Exact conditions for commutativity of projections with crossover.* Tech. Rep. Institute for Adaptive and Neural Computation, University of Edinburgh.

Vose, M. D. (1999). *The simple genetic algorithm: Foundations and theory*. Cambridge, MA: MIT Press.

Vose, M. D., and Whitley, L. D. (1999). A formal language for permutation recombination operators. In *Foundations of Genetic Algorithms (FOGA-5)*, pp. 135–145. San Mateo, CA: Morgan Kaufmann.

Appendix

In the proof of Lemma 8, we have to sum over the set of all functions, the sum containing the product over all elements of Ω. For any function $f : \Omega \to \Omega$ and any $k \in \Omega$, let $S(k, f(k))$ be any expression depending on k and $f(k)$. If we identify the elements of Ω with the set $\{1, 2, \ldots, |\Omega|\}$, then we can identify a function f with the vector whose kth component is $f(k)$. Then

$$\begin{aligned}\sum_f \prod_k S(k, f(k)) &= \sum_{f_1 \in \Omega} \sum_{f_2 \in \Omega} \cdots \sum_{f_{|\Omega|}} \prod_k S(k, f_k) \\ &= \sum_{f_1 \in \Omega} S(1, f_1) \sum_{f_2 \in \Omega} S(2, f_2) \ldots \sum_{f_{|\Omega|}} S\big(|\Omega|, f_{|\Omega|}\big) \\ &= \prod_k \sum_{f_k \in \Omega} S(k, f_k)\end{aligned}$$

which allows the sum and product to be exchanged.