To appear in "Foundations of Genetic Algorithms IV", Morgan Kaufmann (San Mateo), 1996.

Real Representations

Patrick D. Surry[a,b] **& Nicholas J. Radcliffe**[a,b]
{pds,njr}@quadstone.co.uk

[a] Quadstone Ltd, 16 Chester Street, Edinburgh, EH3 7RA, UK
[b] Department of Mathematics, University of Edinburgh, The Kings Buildings, EH9 3JZ, UK

Abstract

This paper introduces two new representations for real-parameter spaces—the Dedekind and Isodedekind representations. Point mutation and uniform crossover—in their generalised, representation-independent form—are shown, when instantiated with respect to these representations, to give rise to familiar operators for continuous domains, such as gaussian mutation, blend crossover and line recombination. Both the Dedekind and Isodedekind representations are highly non-orthogonal (admitting many illegal chromosomes), but, as is demonstrated, this causes no practical or theoretical problems. Moreover, these novel representations are shown to have sensible behaviour as the continuous limit is taken, while both "traditional" binary coding and Gray coding are shown to have pathological behaviour.

1 Introduction

Many optimisation problems are formulated as a search for vectors of real-valued parameters that form extrema of some function. A variety of both local and global techniques with varying degrees of specialisation have been proposed for tackling such problems. Evolutionary algorithms have also been regularly applied in these domains, a particular attraction being that they require only the ability to evaluate the function at any point. Indeed, evolution strategies have been primarily focussed on real-parameter optimisation, with theoretical results specialised to this domain. Traditional genetic algorithms, on the the other hand, are applied to such problems by mapping to a canonical representation space of binary strings for which simple operators are defined. Typical "practical" genetic algorithms specialise these operators by considering the phenotypic effects of the moves they generate in the search domain of real parameters.

In this paper, we demonstrate deep connections between the approaches favoured for continuous domains in evolution strategies and "pragmatic" genetic algorithms, and the operators developed in the "traditional binary" genetic algorithm school for combinatorial optimisation. This is achieved through exploiting a formal procedure for transferring algorithms and operators between arbitrary search domains. The general goal here is to forge a strong link between explicitly stated beliefs about which features of a search domain affect performance and the quality of the instantiated search algorithm. In particular, the aim is that good characterisations and beliefs lead to good search performance, but equally importantly that poor characterisations result in poor search.

We show that by explicitly designing representations that capture beliefs about the structure of the search domain of real parameters (such as the importance of locality and continuity), we can instantiate problem-independent algorithms built from generalised mutation and recombination operators. (In fact, precisely these algorithms have previously been instantiated in combinatorial optimisation domains; Surry & Radcliffe, 1996.) We find that when particular characterisations are employed, we derive commonly used operators such as blend crossover, line recombination and gaussian mutation from the generalised operators.

To facilitate this, we extend previous work on formal construction of representations in discrete (typically combinatorial) search problems to continuous domains. A sequence of representations forming increasingly accurate approximations to the continuous space is employed, and requirements for the limiting process are formulated. When conventional representations for real parameters are considered within this framework, previously suspected peculiarities in their behaviour are confirmed.

We proceed to develop two formal representations for real-parameter evolutionary optimisation based on a formal codification of beliefs about the nature and structure of continuous search domains. We have named the resulting representations the *Dedekind* and *Isodedekind* representations, for reasons that are explained later. We examine the limiting behaviour of these representations, and derive problem-specific forms of generic genetic operators introduced previously. These are seen serendipitously to reduce to 'sensible' operators already widely used for real optimisation.

Having constructed the new representations, we find ourselves able to apply identical (formal) genetic move operators, and therefore identical formal evolutionary *algorithms* with four different representations of real parameter spaces—"traditional" binary coding, Gray coding, Dedekind and Isodedekind. We observe striking qualitative differences in behaviour of these four representations for even the simplest objective functions (figure 1).

The primary purposes of this work are to illuminate deep connections between "evolution strategy style" and "genetic algorithm style" operators, to bridge a gap between discrete and continuous domains, and to expose the formal gene structure underpinning evolutionary approaches to continuous optimisation. It also, however, provides a study in the formal construction of representations and operators from explicit codifications of beliefs about the structure of search domains. In this connection, this work also demonstrates convincingly that the characterisation of a particular problem domain used to induce a representation need not be completely free of 'conflicting' beliefs. Although such characterisations can lead to highly *non-orthogonal* representations (in which the legal values for a given allele are dependent on the current values of others), this is not seen to be problematic in general—indeed the operators derived from such representations may be more powerful than those resulting from simpler ones.

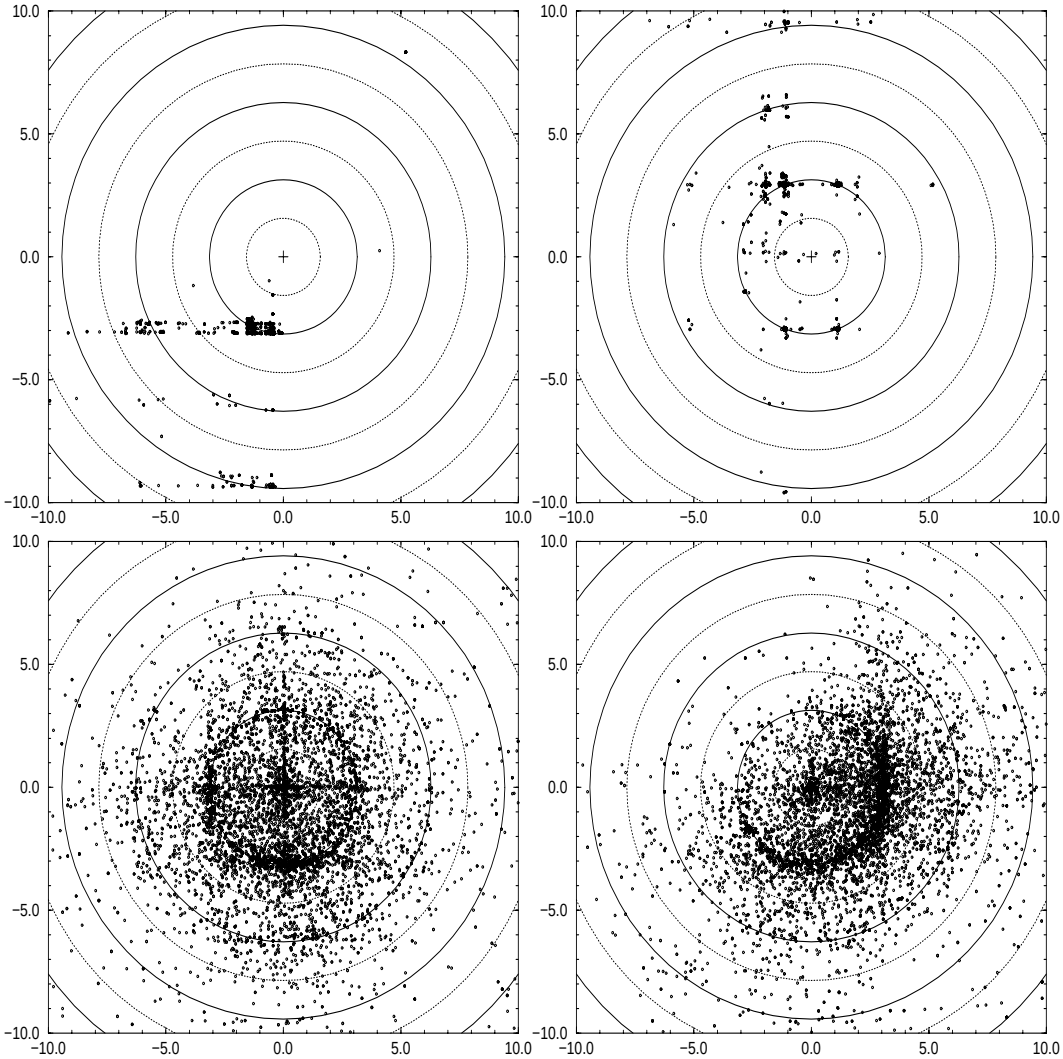

Figure 1: In the spirit of the work presented in Eshelman & Schaffer (1992), this figure illustrates the results of applying an identical formal algorithm instantiated with the four different real-parameter representations discussed herein to Schaffer's F6 function. The two-dimensional function is radially symmetric with global maximum at $(0, 0)$, local maxima on circles of radius $\pi, 2\pi, \ldots$, and local minima on circles of radius $\pi/2, 3\pi/2, \ldots$. The search domain is $[-100, 100] \times [-100, 100]$ of which the figures show the central region. Each figure shows all of the points sampled in one typical run of a fixed algorithm based on the R^3 and BMM operators (see appendix A). Each algorithm sampled approximately 9000 points in the central region during a run of 100 generations with population size 100. The binary representation (top-left) exhibits extremely poor coverage and an extremely striated sampling pattern based on the relative periodicity in the function and the representation. A Gray-coded representation (top-right) typically shows better coverage, as mutation is more effective, but the sampling pattern is clearly biased. The Dedekind representation (bottom-left) shows much better coverage, but since R^3 reduces to BLX-0 there is still a tendency to favour the axis directions, and an inward bias on the population. The algorithm based on the Isodedekind representation (bottom-right) still shows the inward population bias since R^3 reduces to line recombination, but the axial skew is removed.

2 Evolutionary real-parameter optimisation

Optimisation of functions defined over real parameters has a long history, so it is natural that the area has received significant attention from the evolutionary algorithms community. Several now converging schools of thought brought different points of view to the attack.

In the evolution-strategy paradigm (Rechenberg, 1973) and in the evolutionary-programming school (Fogel *et al.*, 1966), the vector of parameters is typically interpreted directly as a "genome", with "gene" values approximated by floating-point machine values. A variety of specialised recombination and mutation operators that directly manipulate these parameters have been employed, but to date their connection to operators used in other search domains has not been apparent. Work with evolution strategies (Baeck & Schwefel, 1993) stresses the importance of gaussian (creep) mutation (possibly with adapted width) as a search operator, based on the so-called *principle of strong causality*, by which small changes in parameter values are assumed to lead to small changes in the objective function. We will see later that by formalising this or other beliefs about the search domain, we can gain insight into what structures the resulting algorithms may be said to be "processing". Recombination methods including line recombination ($\vec{z} = \alpha\vec{x} + (1-\alpha)\vec{y}$), parameter-wise uniform crossover (also known as local-discrete recombination) and blend crossover (local-intermediate recombination) have been employed.

In the genetic-algorithm school, this "parameters-as-genes" approach has not been universally accepted. The common practice has been to represent and manipulate real parameters as fixed-length binary strings using either "traditional" integer coding (where bit strings are decoded as integers and linearly scaled to the appropriate parameter ranges) or "Gray coding" (in which consecutive integers are coded by bit strings that differ in only a single position). Such binary codings allow "standard" genetic operators such as N-point crossover and point mutation to be applied in real-parameter optimisation, albeit with the restriction that the discretisation must result in 2^k points per parameter, for some integer k. They also reflect continuing attachment by many to the dubious *principle of minimal alphabets* (Goldberg, 1989), which has been shown to be motivated by highly questionable theoretical observations (Radcliffe, 1991a; Vose & Liepins, 1991).

An increasing proportion of the genetic algorithms community, however, particularly those working on real-world applications, have pointed out the efficacy of working directly with the real parameters (e.g. Davis, 1991; Michalewicz, 1992). Using "standard" genetic operators in this case—viewing parameters as genes—is problematical as has been pointed out by Goldberg (1990), with the result that *ad hoc* operators have been generally been used, such as "creep mutation" (Davis, 1991) and blend crossover (Eshelman & Schaffer, 1992; generalised from the R^3 operator of Radcliffe, 1991a). Although practically useful, such approaches have lacked a formal basis (for instance, it is not clear what a gene is or how the genetic operators are formally defined), and operators are seen conceptually as acting directly in the search space rather than in a space of genotypes that represents it. In the coming sections we will show that both of the standard evolution-strategy style operators for reals and the standard genetic algorithm operators can be derived from common "representation-independent" template operators. The resulting equivalences are shown in table 1. Figure 2 shows graphically the (phenotypic) effect of the different recombination operators, and figure 3 illustrates the various mutation operators.

In addition to the need for satisfactory mathematical operator derivations, in the case of continuous domains there is a need for clearer understanding of the relationship between operators' effect in a discrete approximation space and in the underlying continuous space. In particular, it seems desirable that operators have well-defined behaviour as the grid spacing shrinks to zero, or at least that we

Evolution strategy term	Genetic algorithm term	Formal derivation
Local-discrete recombination	Uniform crossover	RAR, RTR+ Real
Local-intermediate recombination	BLX-0	RAR, R^3, RTR+ Dedekind
Line recombination	Line recombination	RAR, R^3, RTR+ Isodedekind
n/a	N-point crossover	$\sim$ GNX+ Real
Gaussian mutation	creep mutation	BMM+ Dedekind or Isodedekind
n/a	parameter-wise mutation	BMM+ Real
n/a	bit-wise point mutation	n/a

Table 1: The table summarises several of the operators commonly used in genetic algorithms and evolution strategies (often with different names). Although they may appear to be completely different from one another, they can all be derived as problem-specific forms of generalised problem-independent operators when particular representations are chosen. The "formal" operators RAR, RTR, R^3, GNX and BMM are described in appendix A, BLX-0 is blend crossover with parameter zero, and the Dedekind and Isodedekind representations are described in sections 5.3 and 5.4 respectively.

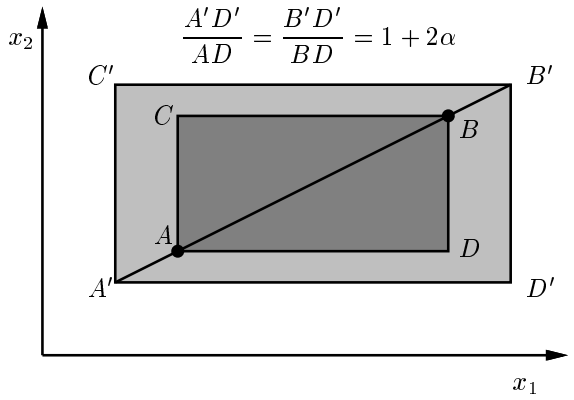

Figure 2: The figure illustrates some of the variety of crossover operators typically used in evolutionary optimisation of real parameters. Here, parents A and B, each a two-component vector, are crossed. Parameter-wise uniform crossover (termed discrete recombination in evolution strategies) generates A, B, C or D with equal probability. For the formal real representation, RAR, RTR, and R^3 are all equivalent to blend crossover with parameter 0 (BLX-0; termed intermediate recombination in evolution strategies), and generate a child uniformly from the rectangle $ACBD$. BLX-α ($\alpha > 0$) generates a child uniformly from the rectangle $A'C'B'D'$. Line recombination generates children uniformly on the line AB, and extended line recombination generates children uniformly on the line $A'B'$. Standard (N-point and uniform) operators with traditional binary codings generate non-localised children which is difficult to show schematically.

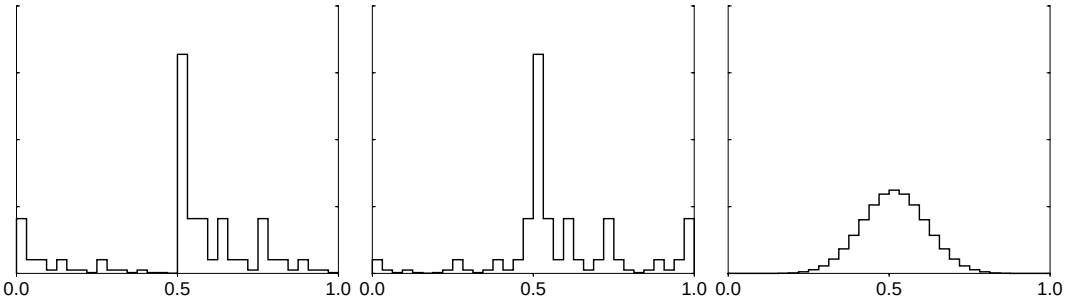

Figure 3: The figure illustrates the effects of instantiating the representation-independent mutation operator BMM (see appendix A) for traditional binary coding (left), Gray coding (center) and the Dedekind representation (right). For the first two representations, BMM reduces to standard point mutation, and in the last case to gaussian creep mutation. For each representation, the real interval $[0, 1]$ has been discretised into 32 points, and the graphs show the probability distribution over possible offspring when the genome representing the point just right of 0.5 is mutated. The pathological distributions in the first two cases help explain some of the behaviour observed in figure 1.

understand what is happening if this is not the case. While gray coding has traditionally been put forward as a "smoother" binary representation for reals, the analysis here will show that it shares with "traditional" binary coding pathological limiting behaviour, while the Dedekind and Isodedekind representations have well-behaved natural continuous limits.

3 Representation in evolutionary search

Evolutionary algorithms are based on genetic operators that manipulate chromosomes. These chromosomes are *representatives* of the structures in the actual search domain, each of which has an associated quality measure. When considering the problem of how to represent a search domain in order to define evolutionary operators, most work has taken one of two approaches. In the first approach, a fixed "canonical" representation is used, and some mapping between the representation space and the search space is constructed. In the second approach, operators are designed specifically for each new search domain, and the problem of representation is largely ignored; conceptually the search operators work directly in the search space. Both of these approaches have significant drawbacks, and we propose an alternative methodology which captures the strengths of both while avoiding their weaknesses.

A canonical representation has the advantage that once operators (and hence algorithms) are defined, they can be applied to any new problem domain—all that is required is to define some mapping from, say, binary strings to the structures in the new search domain. Examples in which both genetic algorithms, which were first conceived for combinatorial problem domains, and evolution strategies, developed for continuous domains, have been coerced into other search domains are relatively commonplace. However, from the point of view of practical optimisation, this has a gigantic drawback. Recent work (Wolpert & Macready, 1995; Radcliffe & Surry, 1995) has confirmed what has been known intuitively for some time—there is no such thing as a free lunch! The representation of the search domain must capture, in some way, the structure of the objective function in order for there to be any possibility of out-performing enumeration. Simply transferring algorithms between problems using a fixed representation space, without explicitly considering the actual representation to be used,

and how structure in the search space is preserved through it, is doomed to failure: no algorithm can be an effective black-box optimiser.

A problem-specific approach avoids these difficulties, as *ad hoc* operators can be defined to exploit known characteristics of the problem at hand, or "standard" operators can be modified by forcing them to make moves that appear "sensible" within the search space. However, this approach has the clear disadvantage that work is not easily transferable to new search domains.

The authors argue that a middle ground is preferable. A formalism has been developed that allows an appropriate representation for a given search domain to be generated directly from statements of belief about the search space. Universal "representation-independent" genetic operators can then be instantiated with respect to that representation. The theory is based on characterising beliefs about the structure of a domain of optimisation problems. This characterisation mathematically generates a representation space and growth function for any given instance of the problem. Once the representation has been chosen, problem-specific forms of any of the generalised genetic operators can be mathematically derived. This is important because many of the representations generated by explicitly characterising beliefs about the structure of the search problem turn out to be *non-orthogonal* (in which not all combinations of alleles are legal), meaning that "traditional" genetic operators can not be used. (Consider, for example, trying to use N-point crossover on two permutations: invalid solutions typically result). This framework enables the separation of algorithm and domain-knowledge to be made completely explicit. Representation-independent search algorithms (constructed with generalised move operators) can be precisely specified mathematically. For a given search domain, beliefs about its structure are mathematically formalised to construct a representation. This representation is then used to instantiate the generalised algorithm to derive a computationally effective, problem-specific search strategy.

It is worth noting that the significance of the choice of representation is *not* primarily the way in which real values are physically stored in a digital computer (which is ultimately always as bit patterns), but rather, the way in which it affects the moves effected in the search space by the chosen genetic operators. Representation, as discussed here, is simply a mathematical device for deriving domain-specific operators that incorporate our explicit beliefs about problem structure.

Forma analysis

The approach taken in this paper is based on *forma analysis* (Radcliffe, 1991a, 1991b, 1994), which is reviewed in appendix A for readers not already familiar with this material.

The basis of forma analysis is that formae (generalised schemata) capture beliefs about problem structure—in particular, that they group solutions of related performance—and that the objective function is to *some* degree separable over them (so that recombination and mutation can be effective). It is then possible to define "representation-independent" operators which manipulate the forma membership properties of solutions so as to respect our beliefs about the problem structure.

Although previous work has focused primarily on finite search domains, such as combinatorial problems like the traveling sales-rep problem (Radcliffe & Surry, 1994), neural network topology optimisation (Radcliffe, 1993), multi-objective pipeline optimisation (Surry *et al.*, 1995) and so forth, some initial work was done on continuous domains (Radcliffe, 1991a, 1991b). In this paper we extend these ideas by considering a limiting sequence of discrete representations. These results are used to define two formal genetic representations for real-parameter optimisation, which are used to derive problem-specific forms of the generalised genetic operators.

Algorithms and search strategies

Once genetic operators have been defined independently of any particular representation or problem domain, it is possible to specify completely problem-independent algorithms—for example, not only can we prescribe the selection methodology, but also the exact recombination and mutation operators the algorithm will use. For any specific problem domain $\mathcal{D}$, we develop a representation using some characterisation χ of our beliefs about its structure, which is used to mathematically derive a problem-specific version of our algorithm. A well-defined mathematical procedure has been developed (Surry & Radcliffe, 1996) which, given a problem instance $I \in \mathcal{D}$ (over a search space $\mathcal{S}$), and a characterisation χ of $\mathcal{D}$, generates a representation space $\mathcal{C}_\chi$ and a growth function $g_\chi : \mathcal{C} \longrightarrow \mathcal{S}$ suitable for application of any of the representation-independent operators described in appendix A (see also figure 4).

4 Scaling properties of discretised representations

Previous work has focused on the representation of discrete spaces. However, we seek to extend our formalism to the case of continuous spaces such as the reals. It is clear that any implementation of a search algorithm using digital computers will be finite, so that we are forced to consider an *approximation* of some kind to the actual search space. It is the goal of this section to formalise the requirements that we might enforce in order to make this approximation meaningful.

First of all, we require that we can (in principle) generate a representation of the continuous search space to any desired level of accuracy. We then require that the actions of our search operators have sensible limiting behaviour as we arbitrarily increase the accuracy of our approximate search space.

Consider a problem domain $\mathcal{D}$ in which each instance I is defined on a continuous search space $\mathcal{S}$, typically a subset of $\mathbb{R}^m$. Suppose further that there is a distance metric, $d : \mathcal{S} \times \mathcal{S} \longrightarrow \mathbb{R}^+$ associated with $\mathcal{S}$. We wish to generate an approximate representation for any given problem instance in the domain, to any degree of accuracy. Let $n \in \mathbb{Z}^+$ indicate the degree of accuracy desired, as formalised below. We suppose that a characterisation, $\chi(I, n)$, is available, which is an automatic procedure for generating a finite representation space $\mathcal{C}_{\chi(I,n)}$ and growth function $g_{\chi(I,n)}$ (which we will abbreviate as $\mathcal{C}_n$ and g_n). The growth function maps chromosomes in the representation space into structures into a finite subset $\mathcal{S}_n$ of the search space, as illustrated in figure 4.

We first require that the approximation of $\mathcal{S}$ can be made arbitrarily good. Formally, we require that for any open subset of $\mathcal{S}$, there is some level of accuracy above which our approximation always represents some point in the subset:

$$\forall B \subseteq \mathcal{S} \ (B \text{ open}) \ \exists n_0 \in \mathbb{Z}^+ \ : \ n > n_0 \Longrightarrow \mathcal{S}_n \cap B \neq \varnothing,$$

where $\mathcal{S}_n = g_n(\mathcal{C}_n)$ is the subset of the search space currently represented.

Furthermore, we require that any search operators to be used exhibit reasonable limiting behaviour. Thus, as we change the degree of accuracy of our approximation, we desire that the action of the operators in the search space does not change radically with respect to the distance function defined on $\mathcal{S}$.

Any search operator, Ω, can be viewed as generating a child chromosome from one or more parent chromosomes and a control parameter selected uniformly from a control set (which provides "randomness"; see Radcliffe, 1994), thus

$$\Omega : \mathcal{C}^q \times \mathcal{K}_\Omega \longrightarrow \mathcal{C},$$

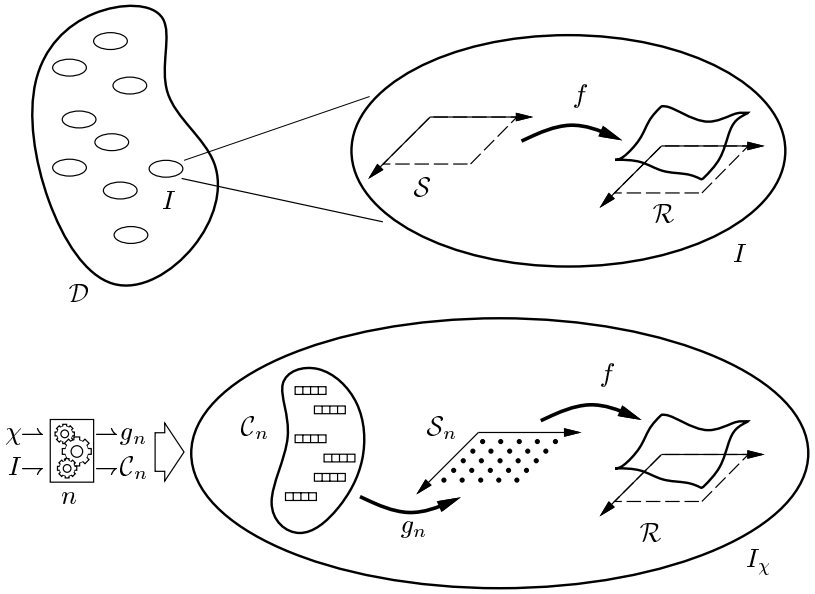

Figure 4: A *problem domain* $\mathcal{D}$ consists of a set of *problem instances*. Each instance I defines a *search space* $\mathcal{S}$ of candidate solutions, a *fitness function* f and a set of objective values $\mathcal{R}$. A *characterisation* χ of the domain specifies a set of equivalences among the solutions for any instance I, for any given accuracy n. These equivalences induce a *representation* made up of a *representation space* $\mathcal{C}_{\chi(I,n)}$ (of *chromosomes*) and a *growth function* $g_{\chi(I,n)}$ mapping chromosomes to a subset $\mathcal{S}_n$ of $\mathcal{S}$. A chromosome x is a string of *alleles*, each of which indicates that x satisfies a particular equivalence on $\mathcal{S}$. Algorithms can be completely specified by their action on the alleles of these generalised chromosomes, making them totally independent of the problem domain itself.

where $\mathcal{K}_\Omega$ is the set of possible control parameters (e.g. cross points, mutation masks, etc.), and q is the arity of the operator (so $q = 1$ for a mutation operator, $q = 2$ for a typical recombination operator, etc.). We can define an equivalent probabilistic form of Ω, which samples uniformly from the control set:

$$\tilde{\Omega}(x_1, \ldots, x_q) \triangleq \Omega(x_1, \ldots, x_q, \kappa) \ : \ \kappa \sim U(\mathcal{K}_\Omega).$$

In order to restrict the limiting behaviour of such an operator, we require that its action on chromosomes representing nearly the same point in the search space tends to be the same. Formally,

$$\forall \varepsilon > 0 \ \forall \{(a_i, x_i)\}_{i=1}^{q} \ (a_i \in \mathcal{S}, x_i \in \mathcal{C}) \ \forall B \subseteq \mathcal{S} \ (B \text{ open}) \exists \delta, p_0 > 0 \ \exists n_0 \in \mathbb{Z}^+ \ :$$
$$n > n_0, \max_{1 \le i \le q} d(g_n(x_i), a_i) < \delta \Longrightarrow |P(g_n(\tilde{\Omega}(x_1, x_2, \ldots, x_q)) \in B) - p_0| < \varepsilon.$$

Thus the probability that the operator generates a representative of a point contained in any fixed open subset (B) of $\mathcal{S}$ must converge to a fixed value (p_0) as the operands (the x_i) converge to representatives of any fixed set of points (the a_i) in $\mathcal{S}$.

These definitions prompt us to redefine the binomial minimal mutation operator, BMM (see appendix A). As originally presented (Radcliffe & Surry, 1994), the operator made a number of minimal mutations chosen from a binomial distribution parameterised by the mutation probability, p_m, and the number of alleles in the genome, n. In order to preserve its behaviour as $n \to \infty$, we advocate that the binomial distribution be parameterised by an *effective* chromosome length, ℓ_n instead of the actual length n. This effective length should be chosen so that a sequence of ℓ_n randomly chosen minimal mutations randomises the chromosome. Typically $\ell_n = n$, but when the representation is highly non-orthogonal (so that the likelihood of "undoing" previous minimal mutations is high), we may have $\ell_n > n$. For example, with the Dedekind representation introduced below, we find that $\ell_n = n^2$.

5 Representations for real-parameter optimisation

In this section, we consider a number of different representations for discretised real-parameter optimisation. We first discuss traditional integer-coding and the related Gray coding, and then go on to develop two new representations and their associated operators based on formally characterising our beliefs about the structure of the search domain.

In generic real-parameter optimisation, the most obvious/general belief we would like a representation to capture is some form of Lipschitz condition (also known as Hölder-continuity)—that small changes in the parameters lead to small changes in the observed function values. Thus, neighbouring solutions in the search space are expected to have related performance. By capturing this idea on an "axis-by-axis" (parameter by parameter) basis, we develop the Dedekind representation, and by considering all possible axis orientations simultaneously we derive the Isodedekind representation. The operators we derive from these representations have been commonly used in evolution strategies, but are now seen to be formally equivalent to operators used in completely different discrete/combinatorial domains.

Traditional approaches to binary coding of real parameters is based (if on any explicit foundation) on the belief that more schemata are better than fewer (the notion of implicit parallelism, giving rise to the principle of minimal alphabets). This has repeatedly been shown to be little more than statistical sleight of hand: one sample is one sample, not many. There is also perhaps some idea that because binary coding "chops up the search space" in many different ways, it lets the algorithm discover useful patterns (Goldberg, 1989; Holland, 1975) but research has shown that is only true if the problem happens to coincide with the particular scaling and location captured in the binary coding. For example, Eshelman & Schaffer (1992) found that simply rescaling or shifting the coordinate axes could dramatically impact performance. (It is reasonable, however, to speculate about constructing a formal representation based on beliefs about periodicity in the objective function—or, indeed, on any other feature thought to be relevant—but this has not as yet been achieved, although an early attempt was made by Radcliffe, 1991a.)

5.1 Approximating the search domain

In evolutionary real-parameter optimisation, we face the problem of representing a discrete approximation to a continuous search space. Here the search space is taken to be a product of real intervals,

$$\mathcal{S} \triangleq \prod_{i=1}^{m} [\alpha_i, \beta_i] \subseteq \mathbb{R}^m.$$

We consider a discretised approximation to the search space generated by the intersection points of a uniform lattice of planes along each co-ordinate axis, namely:

$$\mathcal{S}_n \triangleq \prod_{i=1}^{m} \{\alpha_i, \alpha_i + \Delta_{i,n}, \alpha_i + 2\Delta_{i,n}, \ldots, \beta_i\} \subset \mathcal{S}$$

where

$$\Delta_{i,n} = \frac{\beta_i - \alpha_i}{n-1}.$$

It is thus clear that by using a simple linear transformation, it is sufficient to consider the case $\mathcal{S}_n = \mathbb{Z}_n^m$. (For the traditional binary codings, we can, for simplicity, restrict n to powers of 2, but this is not a significant restriction.)

We begin by considering the case $m = 1$ (in which we represent a single parameter), and examine the traditional binary representations as well as developing the *Dedekind* representation. In section 5.4, we show that building up to higher dimensional search spaces is straightforward, and also introduce the alternative *Isodedekind* representation.

5.2 Traditional genetic algorithm representations

We first examine the traditional methods for coercing a genetic algorithm into a real-parameter optimisation domain. The two standard and variously-championed approaches are traditional integer coding, in which an integer is directly coded in its base 2 representation, and Gray coding, which alleviates one of the perceived problems with the first approach. We present formal definitions of the representations in forma analysis terms, in order that we will later be able to derive forms of generalised genetic operators and to examine their limiting behaviour as we approximate the continuous space more closely.

We will see that neither representation is based on explicitly characterising any particular structure of the underlying optimisation problem, and that this leads to pathological limiting behaviour.

Traditional integer coding

A common approach to representing n adjacent integer values is to use $k = \lceil \log_2 n \rceil$ bits. When such an approach is taken, there is a choice, in principle, of $2^k!$ mappings between the 2^k values and the 2^k strings used to represent them. In practice, almost all such work uses either "traditional" binary coding, in which the ith value is represented by the binary number i, or so-called *Gray coding* (explained below).

With traditional integer-coding, we can write the formal equivalence relations generating the representation as follows:

$$\psi_i(x, y) = \begin{cases} 1, & \text{if } x \otimes 2^i = y \otimes 2^i, \\ 0, & \text{otherwise.} \end{cases}$$

Value	Binary	□□1	□0□	□01	Gray	□□1	□0□	□01	Dedekind	ξ_5^0	ξ_2^1	$\xi_5^0 \cap \xi_2^1$
0	000	-	•	-	000	-	•	-	0000000	•	-	-
1	001	•	•	•	001	•	•	•	0000001	•	-	-
2	010	-	-	-	011	•	-	-	0000011	•	•	•
3	011	•	-	-	010	-	-	-	0000111	•	•	•
4	100	-	•	-	110	-	-	-	0001111	•	•	•
5	101	•	•	•	111	•	-	-	0011111	-	•	-
6	110	-	-	-	101	•	•	•	0111111	-	•	-
7	111	•	-	-	100	-	•	-	1111111	-	•	-

Table 2: The figure shows the way in which elements of $\mathbb{Z}_n$ (used to approximate a real-valued interval) are represented using traditional binary coding, Gray coding, and the Dedekind representation developed here. Several example formae are illustrated in each case. Note that in the first two cases, formae correspond to schemata and are global in extent (including intersections), while in the third case, formae correspond to intervals on the line and encapsulate the notion of locality, presumably important in real optimisation. These representations are used to mathematically derive genetic operators suitable for computational implementation—it is *not* necessary (or even feasible, in the case of Dedekind) to store solutions in the forms shown here.

where $i \in \{0, 1, \ldots, k-1\}$ and $\otimes$ denotes bitwise-and. These equivalence relations induce the basic formae (alleles) $\xi_0^0, \xi_0^1, \ldots, \xi_i^0, \xi_i^1, \ldots, \xi_{k-1}^0, \xi_{k-1}^1$ which we can identify exactly with schemata

$$\begin{aligned} \xi_i^0 &\equiv \square \cdots \square 0 \square \cdots \square \\ \xi_i^1 &\equiv \square \cdots \square 1 \square \cdots \square \end{aligned}$$

with the 0 or 1 in the ith position. It is clear that through intersections of these basic formae we can build all higher-order formae (again, exactly the higher-order schemata), and can identify individual solutions by noting the equivalence class to which they belong for each of the basic equivalence relations. Examples of this coding indicating the membership patterns of several formae are shown in table 2. Qualitatively we see that formae are neither local nor purely periodic in extent. Further, it is clear that as we increase the number of formae in order to approximate $\mathcal{S}$ more and more accurately, there is no clear notion of the limiting properties of the formae. This will be made evident in section 5.6 when we examine the limiting behaviour of the genetic operators derived from this representation.

Gray coding

The principal motivation for considering Gray coding is the perceived problem of Hamming Cliffs with traditional integer coding. An example of a Hamming cliff is the transition from 7 to 8 in the traditional coding, where a relatively large number of bits change value with a small step in the search space (e.g. $\cdots 0111$ to $\cdots 1000$). The attraction of Gray coding (Caruana & Schaffer, 1988) is that the strings representing adjacent values always differ by exactly one bit. There are numerous possible mappings of the integers to binary strings that have this adjacency property, but the most commonly used one codes the integer x as the (binary) value of $x \oplus \lfloor x/2 \rfloor$ where $\oplus$ is the bitwise exclusive-or operator. If we write the binary value of x (the traditional integer coding) as $b_{k-1} \cdots b_1 b_0$ and the Gray-coded representation of i as $g_{k-1} \cdots g_1 g_0$ then we have the relationships

$$g_i = b_{i+1} \oplus b_i$$

and

$$b_i = b_{i+1} \oplus g_i,$$

which allow conversion from one form to the other (taking $b_k = 0$).

These relationships allow us to write the equivalence relations which define the Gray-coding representation:

$$\psi_i(x,y) = \begin{cases} 1, & \text{if } (x \oplus \lfloor x/2 \rfloor) \otimes 2^i = (y \oplus \lfloor y/2 \rfloor) \otimes 2^i, \\ 0, & \text{otherwise.} \end{cases}$$

where $i \in \{0, 1, \ldots, k-1\}$. As with traditional integer coding, the formae can be exactly identified with (now Gray-coded) schemata. An example of the coding scheme along with the members of several formae are shown in table 2. Although these formae never contain 'singleton' solutions, it is still clear that they are still highly non-local in extent. It will also be shown that the limiting properties of the genetic operators (as the level of approximation is improved) are no better than the traditional coding.

5.3 Representations that "capture" continuity

The Dedekind representation

We seek to define a representation for $\mathbb{Z}_n$ based on our beliefs about the structure of the problems we will be attacking. For the wide class of real-parameter optimisation problems, perhaps the simplest belief we might hold is some notion of continuity. That is, we believe that small changes in the parameter values will lead to small changes in the objective function. In evolution strategies, this belief is termed *the principle of strong causality*, and in real analysis functions with such a property are termed *Hölder continuous* or *Lipschitz*.

In order to quantify this belief, we seek to characterise groups of solutions (formae) with related performance. To do this, we will define equivalence relations that capture locality, based on the idea of *Dedekind cuts*:

> A Dedekind cut is a partitioning of the rational numbers into two non-empty sets, such that all the members of one are less than all those of the other. For example, the positive irrationals can be defined as Dedekind cuts on the positive rationals
> (e.g. $\sqrt{2} \triangleq \langle \{x \in \mathbb{Q}^+ \mid x^2 < 2\}, \{x \in \mathbb{Q}^+ \mid x^2 > 2\} \rangle$).

Thus the basic equivalence relations we define on $\mathbb{Z}_n$ are cuts

$$\psi_i(x,y) = \begin{cases} 1, & \text{if } x, y \geq i \text{ or } x, y < i, \\ 0, & \text{otherwise,} \end{cases}$$

where $i \in \{1, \ldots, n-1\}$. These equivalence relations induce half-space equivalence classes of the following type:

$$\begin{aligned} \xi_i^0 &\equiv \{0, 1, \ldots, i-1\}, \\ \xi_i^1 &\equiv \{i, i+1, \ldots, n-1\}. \end{aligned}$$

It is easy to see that intersections of these basic formae result in sets defining closed intervals, as illustrated in table 2. Note that, formally, this representation has $n-1$ highly non-orthogonal (constrained) binary genes coding a single approximated parameter, instead of the $k = \lceil \log_2 n \rceil$ orthogonal genes

of traditional integer coding or Gray coding. However, we will see that the operators we derive from this representation and their limiting behaviour as we increase the level of approximation ($n \to \infty$) are much more natural with our new definitions.

While it is somewhat ironic that this formalism of the real representation utilises binary genes, we emphasise once again that we do *not* propose to store or manipulate solutions in this form, but only to apply our design principles to develop and analyse our genetic operators.

5.4 Extending the representations to multiple parameters

It is a simple matter to extend any of the representations to higher dimensions by forming products of the one-dimensional equivalence relations. For example, in a two-dimensional search space, approximated by $\mathbb{Z}_n \times \mathbb{Z}_n$, the basic equivalence relations for the Dedekind representation are of the form

$$\psi_{ij}(x,y) \equiv \psi_i(x_1,y_1) \times \psi_j(x_2,y_2) = \begin{cases} 1, & \text{if } (x_1, y_1 \geq i \text{ or } x_1, y_1 < i) \\ & \text{and } (x_2, y_2 \geq j \text{ or } x_2, y_2 < j), \\ 0, & \text{otherwise,} \end{cases}$$

with equivalence classes of the form

$$\begin{aligned}
\xi_{ij}^{00} \equiv \xi_i^0 \times \xi_j^0 &= \{(0,0), \ldots, (0,j-1), \ldots, (i-1,0), \ldots, (i-1,j-1)\}, \\
\xi_{ij}^{10} \equiv \xi_i^1 \times \xi_j^0 &= \{(i,0), \ldots, (i,j-1), \ldots, (n-1,0), \ldots, (n-1,j-1)\}, \\
\xi_{ij}^{01} \equiv \xi_i^0 \times \xi_j^1 &= \{(0,j), \ldots, (0,n-1), \ldots, (i-1,j), \ldots, (i-1,n-1)\}, \\
\xi_{ij}^{11} \equiv \xi_i^1 \times \xi_j^1 &= \{(i,j), \ldots, (i,n-1), \ldots, (n-1,j), \ldots, (n-1,n-1)\}.
\end{aligned}$$

The traditional binary and Gray codings can be similarly extended.

The Isodedekind representation

An alternative approach to extending the Dedekind representation to multiple parameters is to define new equivalence relations. Because it seems somewhat artificial to characterise locality only along the axis directions, we might think of devising a characterisation to capture locality more generally. Thus, we could define basic equivalence relations which partition $\mathbb{Z}_n^m$ using cut planes which have arbitrary orientation. This would require our conceptual chromosome to have alleles indicating on which side of a series of cuts it fell in each possible direction from the origin. Thus in the limiting case, the chromosome consists of a continuous infinity of continuous infinities of genes! We do not exhibit a precise discrete formulation here for reasons of space, but it is straightforward to visualise the limiting forms that the generalised operators take in such a case (see for example figure 5). The Isodedekind representation demonstrates convincingly that even extremely non-orthogonal characterisations can be used successfully.

5.5 Forma variance calculations

One measure which indicates how well formae succeed in grouping together solutions of related fitness is mean forma variance. By generating random formae of a particular size and measuring the fitness variance within them, we can estimate the mean variance for formae of a given size. This was shown to be a good qualitative indicator of relative algorithmic performance (Radcliffe & Surry, 1994).

For the Dedekind and Isodedekind representation, all formae can be represented as convex simplices in $\mathbb{R}^m$. For illustrative purposes, we consider the one-dimensional case in which formae become intervals (although it is not difficult to extend the results to higher dimensions). It is then a simple matter to write down an expression for the fitness variance of a forma (interval) of size τ, centered at $x = t$, with objective function $f(x)$. Namely,

$$\sigma^2(t,\tau) = \frac{1}{\tau}\int_{t-\tau/2}^{t+\tau/2}(f(x) - \bar{f}(t,\tau))^2 dx,$$

where

$$\bar{f}(t,\tau) = \frac{1}{\tau}\int_{t-\tau/2}^{t+\tau/2} f(x)dx.$$

It is straightforward to show that requiring the forma variance to decrease to zero as forma size (τ) decreases is closely related to making a Lipschitz assumption on the objective function.

For instance, assume that f is Lipschitz over an interval $[\alpha, \beta]$, that is,

$$x, y \in [\alpha, \beta] \Longrightarrow |f(x) - f(y)| < \varepsilon|x - y|$$

for some $\varepsilon \in \mathbb{R}^+$. Now consider any forma of size τ contained in $[\alpha, \beta]$. By the mean value theorem, we know that $\bar{f}(t,\tau) = f(c)$ for some $c \in [t - \tau/2, t + \tau/2]$. We then have,

$$\begin{aligned}
\sigma^2(t,\tau) &< \frac{1}{\tau}\int_{t-\tau/2}^{t+\tau/2}(\varepsilon(x-c))^2 dx \\
&= \frac{\varepsilon^2}{3\tau}(x-c)^3\Big|_{t-\tau/2}^{t+\tau/2} \\
&\leq \frac{\varepsilon^2\tau^2}{3}
\end{aligned}$$

Conversely, it is trivial to show that if f is nowhere Lipschitz on $[\alpha, \beta]$, that

$$\sigma^2(t,\tau) \geq \frac{\varepsilon^2\tau^2}{12}$$

Thus the notion that formae should group together solutions of related fitness has been shown to be closely linked to the characterisations on which we based the construction of the Dedekind and Isodedekind representations—namely that the functions of interest were in some sense smooth (satisfying a Lipschitz condition, or more loosely the principle of strong causality).

For the traditional integer coding and Gray codings used with genetic algorithms, there is no analogous limiting behaviour that can be extracted. We might speculate on some relationship to periodic behaviour of the function, but it is not simply that. It has been "discovered" several times that simply by shifting the origin or rescaling the axes, the behaviour of algorithms based on these representation can change radically. This is clearly very undesirable behaviour.

Fundamentally, we argue that this stems from the lack of a principled foundation for the traditional representations—they do not encapsulate particular beliefs about the problem domains of interest. In fact, many workers have taken the diametrically opposite approach of trying to discover what it is

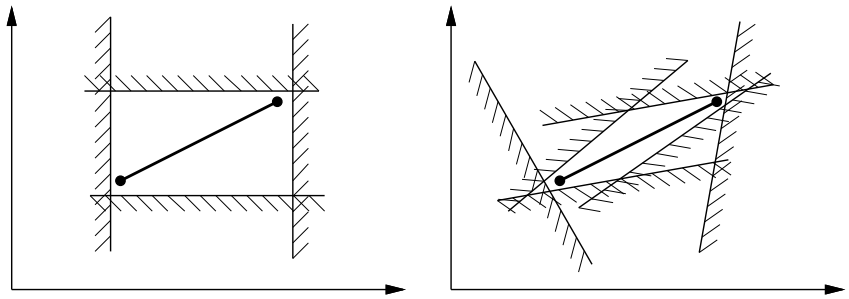

Figure 5: The figure illustrates two problem-specific forms of the random respectful recombination (R^3) operator for a two-dimensional real-parameter optimisation problem. (a) On the left, the Dedekind representation is defined by cut formae along each axis. Since R^3 requires that the child be a member of all formae to which both parents belong, the child must lie within the hypercube (square here) with its two parents at opposite vertices. Hence R^3 reduces to BLX-0 in this case. (b) On the right, the Isodedekind representation is defined by cut formae in arbitrary directions. Thus R^3 requires, in the limit, that the child lies on the line segment bounded by the two parents, so that R^3 is equivalent to line recombination in this case. It is also instructive to compare algorithms based on these different representations which incorporate R^3—see figure 1.

that these representations *are* characterising, for instance by constructing specialised functions over the integers which lead the algorithms in particular directions. While such research may be intrinsically interesting, it is far from clear that it bears on the problem of how algorithms based on these representations behave for "real" functions.

5.6 Derivation of operators

It is now straightforward to derive forms of the generalised genetic operators described in appendix A. The derivations are summarised in table 1. These operators can then be used to instantiate a fixed representation-independent algorithm for each of the four representations presented here. This is illustrated for a simple objective function in figure 1.

For the traditional binary and Gray codings, the generalised operators reduce to "standard" forms, since the representations are orthogonal (all combination of allele values are legal). Thus, RAR, RTR and R^3 reduce to uniform crossover, GNX reduces to N-point crossover, and BMM reduces to bitwise point mutation.

For the Dedekind representation, it is clear that both R^3 and RTR require that the child be uniformly selected to lie in the box determined by the parents (see also figure 5). Thus, in one dimension, the crossover operator $X : \mathbb{Z}_n \times \mathbb{Z}_n \times [0,1) \longrightarrow \mathbb{Z}_n$ acting on parents x and y (without loss of generality, $x < y$) and control parameter $\kappa \in [0,1)$ results in the child $X(x,y,\kappa) = x + \lfloor \kappa(y-x+1) \rfloor$. It is clear that in the limit of $n \to \infty$, this is equivalent to BLX-0 as defined by Eshelman & Schaffer (1992); see figure 2. For this representation, it is not difficult to see that RAR is also equivalent to BLX-0 (it is easy to show that the child must lie in the interval defined by the parents, and only slightly more difficult to demonstrate that the likelihood is uniform over the interval).

For the Isodedekind representation, R^3 and RTR reduce to line recombination as shown in figure 5, and RAR simply requires that we are able to generate any point in the search space.

Turning to mutation, if we analyse our representation-independent mutation operator BMM with the Dedekind representation, it is clear that a minimal mutation involves flipping the value of one of the two bits forming the transition from ones to zeros in the genome. Thus a fixed-length sequence of minimal mutations is equivalent to a random walk away from the original transition point. We will show that this reduces in the limit of $n \to \infty$ to standard gaussian creep mutation with width parameterised by the gene-wise mutation probability.

Proof: Assume that each parameter is represented by n genes in the Dedekind representation. Consider the action of BMM on a single parameter (as it is clear that genes in different parameters are orthogonal). Given a gene-wise mutation rate p_m, and an effective chromosome length ℓ_n (which we will find must be proportional to n^2), we make a binomial number s of minimal mutations, where s is chosen from the distribution $S \sim B(\ell_n, p_m)$. Now, we know that the binomial distribution $B(n,p)$ is asymptotically approximated by the normal distribution $N(np, \sqrt{np(1-p)})$ provided that $\sqrt{np(1-p)} \ll np$. In our case, we take p_m fixed, and $\ell_n \to \infty$ as $n \to \infty$ so that S is asymptotically approximated by $S \sim N(\ell_n p_m, \sqrt{\ell_n p_m (1-p_m)})$. Now, each minimal mutation results in a step of size $\Delta = L/n$ to the left or right of the current value, where L is the length of the interval in which the parameter lies. Thus once we have chosen a number of steps s, distributed according to $S \sim N(\ell_n p_m, \sqrt{\ell_n p_m (1-p_m)})$ we perform a random walk with step length Δ, so that the final displacement x is distributed conditionally on s, according to the normal distribution $X|S \sim N(0, \sqrt{s}L/n)$. Further, we can thus write the unconditional p.d.f. for X as:

$$p(x) = \int_{-\infty}^{\infty} p(x|s)p(s)ds.$$

Consider now the moment-generating function for X, $m_X(t)$, given by:

$$\begin{aligned} m_X(t) = E(e^{tx}) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p(x|s)p(s)e^{tx}dsdx \\ &= \int_{-\infty}^{\infty} p(s)\int_{-\infty}^{\infty} p(x|s)e^{tx}dxds \\ &= \int_{-\infty}^{\infty} p(s)m_{X|S}(t)ds. \end{aligned}$$

Now, since $X|S \sim N(0, \sqrt{s}L/n)$, a normal distribution, we know that $m_{X|S}(t) = \exp(0t + (\sqrt{s}L/n)^2 t^2/2)$, which yields

$$\begin{aligned} m_X(t) &= \int_{-\infty}^{\infty} p(s)\exp\left(\frac{s}{2}\left(\frac{Lt}{n}\right)^2\right)ds \\ &= m_S\left(\frac{1}{2}\left(\frac{Lt}{n}\right)^2\right). \end{aligned}$$

Finally, since $S \sim N(\ell_n p_m, \sqrt{\ell_n p_m(1-p_m)})$ we have $m_S(t) = \exp(\ell_n p_m t + \ell_n p_m(1-p_m)t^2/2)$, so that

$$m_X(t) = \exp\left(\frac{p_m L^2 \ell_n}{2n^2}t^2 + \frac{p_m(1-p_m)L^4\ell_n}{8n^4}t^4\right).$$

We thus choose our length scale $\ell_n = n^2$ to make the limit finite, and see that $\lim_{n\to\infty} m_X(t) = \exp(p_m L^2 t^2/2)$. This is simply the moment generating function for $N(0, \sqrt{p_m}L)$ so the final displacement X must have the identical distribution. Thus BMM instantiated for the Dedekind representation is that operator which adds gaussian noise with width $\sqrt{p}_m L$ to each parameter.

Note that for the Isodedekind representation a sequence of minimal mutations involves a random walk with steps taken in arbitrary directions in $\mathbb{R}^m$. Thus BMM is likely to be equivalent to Gaussian mutation, but the precise scaling factors have not yet been derived.

6 Discussion

This paper has presented formal constructions for two genetic representations for real-parameter optimisation based on characterising the notion of locality—the Dedekind and Isodedekind representations. Generalised genetic operators are shown to reduce in these representations to standard forms, namely blend crossover, line recombination and gaussian creep mutation which are widely used in practice. Both of these representations are highly non-orthogonal, but this is seen not to present difficulties. Analysis of the limiting behaviour of discrete approximations to a continuous search space has shown that the more traditional binary and Gray codings have pathological behaviour. This is illustrated qualitatively when a fixed algorithm is instantiated for all four representations and radically different behaviour is observed (figure 1).

A Forma analysis

Holland (1975) identified subsets of a search space of strings using *schemata*—sets of strings that share certain gene values. His schema theorem shows how the *observed* fitness of any schema in a population can be used to bound the *expected* instantiation of the same schema in the next generation, under the action of proportional selection. Several authors have generalised the idea of schema and shown that the theorem applies to arbitrary subsets of the search space, provided that suitable disruption coefficients are chosen (Radcliffe, 1991a; Vose & Liepins, 1991).

In Radcliffe's work, subsets of the search space, $\mathcal{S}$, are identified as *formae*. Typically, the formae are defined as the equivalence classes induced by a set of equivalence relations, Ψ. (An equivalence relation $\psi : \mathcal{S} \times \mathcal{S} \longrightarrow \{0, 1\}$ can be thought of as a function which returns 1 if a given pair of solutions are "equivalent" and 0 otherwise.) Any solution can then be identified by specifying the equivalence class to which it belongs for each of the equivalence relations. Loosely speaking, we identify genes with a set of basic equivalence relations (from which any member of Ψ can be constructed) and alleles with the corresponding equivalence classes. For instance, "same hair colour" and "same eye colour" might be two basic equivalence relations in Ψ, which would induce the formae "red hair", "brown hair", "blue eyes", etc. Higher order formae are then constructed by intersection, e.g. "brown hair and blue eyes".

The selection of an appropriate set of equivalence relations for a particular class of problem is an open problem. We assume that domain knowledge *characterised* by an algorithmic procedure which generates the required equivalence classes for any given problem instance. For example, in the travelling sales-rep problem, we might reasonably believe that tours sharing any given edge will have related performance (indeed this is clear in this case since the length of a tour is simply the sum of its edge lengths). Thus we might propose a characterisation which generates formae based on equivalence relations of the form 'has edge $\overline{xy}$' (although other characterisations are also possible; see Radcliffe & Surry, 1994).

However, several ideas have been previously proposed (Radcliffe, 1991a) which summarise the way formae should partition the search space. These permit the construction of representation-independent operators that manipulate solutions in effective ways.

- Correlation within formae. The formae should group together solutions of related fitness. One measure of this is the mean fitness variance over formae of a given size. This has been measured experimentally for four representations in the travelling sales-rep problem and shown to correlate well with performance (Radcliffe & Surry, 1994).
- Minimal degeneracy[1]. The number of genomes representing each member of $\mathcal{S}$ should be small.
- Computability. It must be possible to efficiently exhibit members of any given formae computationally (clearly equivalence relations based on tour length in the TSP could be mathematically specified, but it would be computationally infeasible to generate members of particular formae).

These restrictions have not been investigated in depth, but have not been a problem with those representations investigated to date.

Representation-independent operators

The way in which formae are thought to group solutions of related performance suggests several *design principles* for constructing genetic operators. Such operators can be precisely defined independently of any particular representation, by specifying how they manipulate the formae-membership properties of their operand(s). For any given representation, we can use the definition of the operators to formally derive a problem-specific version of that operator. This leads in some cases to previously known operators in a given search domain, but in other cases leads to new insights about what form of recombination or mutation might be applicable to a given problem domain. The generalised operators are particularly relevant to non-orthogonal representations (in which not all combinations of alleles are valid), which often arise when the problem characterisation is based on beliefs which are not completely compatible. A number of these principles and related operators are summarised below.

Respect Respect requires that children are members of all formae to which both their parents belong. For example, if there were equivalence relations about hair colour and eye colour in Ψ, then if both parents had red hair and green eyes, so should all children produced by X.

More formally, a recombination operator $X : \mathcal{S} \times \mathcal{S} \times \mathcal{K}_X \longrightarrow \mathcal{S}$ (where $\mathcal{K}_X$ is a set of control parameters such as cross-points or crossover masks) is said to *respect* the set of formae Ξ generated by Ψ iff

$$\forall \xi \in \Xi \, \forall a \in \xi \, \forall b \in \xi \, \forall \kappa \in \mathcal{K}_X : \; X(a, b, \kappa) \in \xi.$$

Random respectful recombination (R^3) is defined as that operator which selects a child uniformly at random from the set of all solutions which share all characteristics possessed by both parents (their *similarity set*).

Transmission A recombination operator is said to be *strictly transmitting* if every child it produces is equivalent to one of its parents under each of the basic equivalence relations (loosely, every gene is set to an allele which is taken from one or other parent). Thus, if one parent had red hair and the other had brown hair, then transmission would require that the child had either red or brown hair.

[1] The term *redundancy* has previously been used to mean the same thing, but we now urge that redundancy should be reserved for the situation in which chromosomes contain more information than strictly necessary to specify the solution they represent.

The *random transmitting recombination* (RTR) operator is defined as that operator which selects a child uniformly at random from the set of all solutions belonging only to basic formae present in either of the parents (their *dynastic span*).

Assortment Assortment requires that a recombination operator be capable of generating a child with any compatible characteristics taken from the two parents. In our example above, if one parent had green eyes and the other had red hair, and if those two characteristics are compatible, assortment would require that we could generate a child with green eyes and red hair.

Formally, a recombination operator is said to *properly assort* the formae generated by Ψ iff

$$\forall \xi_1, \xi_2 \in \Xi \, (\xi_1 \cap \xi_2 \neq \varnothing) \; \forall a_1 \in \xi_1 \; \forall a_2 \in \xi_2 \; \exists \kappa \in \mathcal{K}_X : \; X(a_1, a_2, \kappa) \in \xi_1 \cap \xi_2.$$

The *random assorting recombination operator* (RAR), a generalised form of uniform crossover, has been previously defined (Radcliffe, 1992). It proceeds by placing all alleles from both parents in a conceptual bag (possibly with different weights), and then repeatedly draws out alleles for insertion into the child, discarding them if they are incompatible with those already there. If the bag empties before the child is complete, which can happen if not all combinations of gene values are allowed (so that the representation is *non-orthogonal*) remaining genes are set to random values that are compatible with those genes already set.

A *generalised N-point crossover*, GNX, has also been proposed (Radcliffe & Surry, 1994). This proceeds in much the same way as standard N-point crossover, dividing the two parents with N cut-points, and then using genetic material from alternating segments. The alleles within each segment are tested in a random order for inclusion in the child, and any remaining gaps are patched by randomly selecting compatible alleles first from the unused alleles in the parents, and then from all possible alleles.

Ergodicity This demands that we select operators such that it is possible to move from any location in the search space to any other by their repeated action. (Typically a standard mutation operator is sufficient.)

Binomial minimal mutation, BMM, a generalisation of standard point-wise mutation, has been proposed in Radcliffe & Surry (1994). Minimal mutations are defined to be those moves which change the fewest possible number of alleles in a solution (in non-orthogonal representations it may be necessary to change more than one allele at a time to maintain legality). BMM performs a binomially-distributed number (parameterised by the genome length and a gene-wise mutation probability) of minimal mutations, and does not forbid mutations which 'undo' previous ones.

Work is also ongoing which investigates the ramifications of a principle of *disdain* in which formae non-membership is used to compare parent solutions. This leads to the development of search operators such as directed mutation and simplex-like recombination.

References

T. Bäck and H.-P. Schwefel, 1993. An overview of evolutionary algorithms for parameter optimisation. *Evolutionary Computation*, 1(1):1–24.

R. A. Caruana and J. D. Schaffer, 1988. Representation and hidden bias: Gray vs. binary coding for genetic algorithms. In *Proceedings of the 5th International Conference on Machine Learning*. Morgan Kaufmann (Los Altos).

L. Davis, 1991. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold (New York).

L. J. Eshelman and D. J. Schaffer, 1992. Real-coded genetic algorithms and interval schemata. In D. Whitley, editor, *Foundations of Genetic Algorithms 2*. Morgan Kaufmann (San Mateo, CA).

L. J. Fogel, A. J. Owens, and M. J. Walsh, 1966. *Artificial Intelligence Through Simulated Evolution*. Wiley Publishing (New York).

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

D. E. Goldberg, 1990. Real-coded genetic algorithms, virtual alphabets, and blocking. Technical Report IlliGAL Report No. 90001, Department of General Engineering, University of Illinois at Urbana-Champaign.

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

Z. Michalewicz, 1992. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer Verlag (Berlin).

N. J. Radcliffe and P. D. Surry, 1994. Fitness variance of formae and performance prediction. In L. D. Whitley and M. D. Vose, editors, *Foundations of Genetic Algorithms III*, pages 51–72. Morgan Kaufmann (San Mateo, CA).

N. J. Radcliffe and P. D. Surry, 1995. Fundamental limitations on search algorithms: Evolutionary computing in perspective. In J. van Leeuwen, editor, *Computer Science Today: Recent Trends and Developments, Lecture Notes in Computer Science, Volume 1000*, pages 275–291. Springer-Verlag (New York).

N. J. Radcliffe, 1991a. Equivalence class analysis of genetic algorithms. *Complex Systems*, 5(2):183–205.

N. J. Radcliffe, 1991b. Forma analysis and random respectful recombination. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 222–229. Morgan Kaufmann (San Mateo).

N. J. Radcliffe, 1992. Genetic set recombination. In D. Whitley, editor, *Foundations of Genetic Algorithms 2*. Morgan Kaufmann (San Mateo, CA).

N. J. Radcliffe, 1993. Genetic set recombination and its application to neural network topology optimisation. *Neural Computing and Applications*, 1(1):67–90.

N. J. Radcliffe, 1994. The algebra of genetic algorithms. *Annals of Maths and Artificial Intelligence*, 10:339–384.

I. Rechenberg, 1973. *Evolutionstrategie—Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog (Stuttgart).

P. D. Surry and N. J. Radcliffe, 1996. Formal algorithms + formal representations = search strategies. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H. Schwefel, editors, *to appear in Parallel Problem Solving from Nature IV*, pages 366–375. (Springer-Verlag, LNCS 1141).

P. D. Surry, N. J. Radcliffe, and I. D. Boyd, 1995. A multi-objective approach to constrained optimisation of gas supply networks: The COMOGA method. In T. C. Fogarty, editor, *Evolutionary Computing: AISB Workshop*, pages 166–180. Springer-Verlag, Lecture Notes in Computer Science 993.

M. D. Vose and G. E. Liepins, 1991. Schema disruption. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 237–243. Morgan Kaufmann (San Mateo).

D. H. Wolpert and W. G. Macready, 1995. No free lunch theorems for search. Technical Report SFI–TR–95–02–010, Santa Fe Institute.