Mapping the Royal Road and other Hierarchical Functions

Janet Wiles j.wiles@itee.uq.edu.au
School of Psychology *and*
School of Information Technology and Electrical Engineering
University of Queensland
Queensland, 4072
Australia

Bradley Tonkes btonkes@itee.uq.edu.au
School of Information Technology and Electrical Engineering
University of Queensland
Queensland, 4072
Australia

Abstract

In this paper we present a technique for visualising hierarchical and symmetric, multimodal fitness functions that have been investigated in the evolutionary computation literature. The focus of this technique is on landscapes in moderate-dimensional, binary spaces (i.e., fitness functions defined over $\{0,1\}^n$, for $n \leq 16$). The visualisation approach involves an unfolding of the hyperspace into a two-dimensional graph, whose layout represents the topology of the space using a recursive relationship, and whose shading defines the shape of the cost surface defined on the space. Using this technique we present case-study explorations of three fitness functions: royal road, hierarchical-if-and-only-if (H-IFF), and hierarchically decomposable functions (HDF). The visualisation approach provides an insight into the properties of these functions, particularly with respect to the size and shape of the basins of attraction around each of the local optima.

Keywords

Visualisation, royal road, hierarchical if-and-only-if, hierarchically decomposable functions, fitness landscape.

1 Introduction

A variety of techniques exist for studying the structure and properties of high-dimensional landscapes. Such approaches can provide insights for selecting appropriate optimisation processes. The No Free Lunch theorem asserts the equality of all search processes when averaged over all cost surfaces (Wolpert and Macready, 1997). However, it is arguable that not all cost surfaces are equally likely, so that some optimisation processes will be more effective than others *in practice* (Christensen and Oppacher, 2001). For efficient optimisation, an algorithm (or its parameters) should be tailored to the features of the task.

For problems that are of practical interest, it remains an open question as to how to determine which optimisation algorithm to employ. One approach has been to compare the empirical performance of a variety of algorithms on a suite of test functions (Belding, 2001). Much early work in this area used example 'real-world' functions as benchmarks and operated on the implicit premise that the benchmark and target functions had 'real-worldness' in common. More recent approaches have considered hand-constructed, parameterisable functions with known and tunable characteristics. For this approach to be informative about novel cost surfaces, there must be some principle suggesting appropriate similarities between the benchmark problem and the novel problem under consideration. Knowing the metric that should be used to judge the similarity of two cost surfaces requires an understanding of why an algorithm performs well on a particular benchmark problem.

One barrier to understanding why an algorithm is effective is understanding the important features of the search landscape. For 'interesting' problems it is computationally too expensive to compute the entire surface, so one approach is to describe the surface with a set of estimated statistics, such as the number of local optima, and their relative distances apart. Such explorations of cost surfaces have been used for describing NK landscapes (Kauffman, 1993).

An alternative approach to understanding cost surfaces, applicable to smaller benchmark problems, is the use of visualisation techniques. Visualising cost surfaces for problems of more than a few dimensions is a difficult task for the human perceptual system. Unfortunately, as is well known, landscapes in 2- or 3-D, which are most intuitive for humans, can be misleading with respect to properties of higher dimensional spaces. Ideally a test problem should be small enough to be efficiently computed, while remaining complex enough to provide insights into scaled-up versions.

A wide variety of techniques exist for visualising multivariate data (see Tufte, 1992 for a classic guide). These include dimension-reduction techniques such as PCA and multi-variate scaling (e.g., Friedman and Tukey, 1974), graph visualisation (e.g., Pryke, 1996), and animation (e.g., Buja and Asimov, 1986). Specialist techniques have also been developed in many domains. For example, in neural network research, methods include direct representation of high dimensional weight spaces, such as Hinton diagrams (Hinton et al., 1984), and projections to lower dimensions such as hyperplane animators for activation spaces (e.g., Munro, 1992), as well as a variety of other dimension reduction techniques (e.g., Wiles and Bloesch, 1992).

For visualising high-dimensional discrete data of a moderate number of dimensions ($3 \leq n \leq 20$) one family of techniques has been widely employed. It involves a recursive layout process in which the high-dimensional space is unfolded on to a two dimensional space so that the high-dimensional topology retains a systematic relationship in the low-dimensional representation. The technique is described in detail in Section 2.

This technique has been independently proposed many times (Collins, 1997), being variously named 'quadcodes' (Shine and Eick, 1997), 'hierarchical axes' (Mihalisin et al., 1991), 'dimensional stacking' (LeBlanc et al., 1991), and the 'search space matrix' (Collins, 1997) and bears a close resemblance to Karnaugh maps, a standard technique from circuit theory for simplifying the design of digital circuits (Hill and Peterson, 1968). The latter studies have either considered distributions of populations in genetic algorithms (thus focussing on multivariate data distributions rather than high-dimensional surfaces) or have not probed the properties of any functions in great detail, considering only simple functions. In circuit theory, the technique was used to visualise boolean functions of typically two to six variables and was regarded as a pictorial form of a truth table. Circuits could be visualised in terms of the union and intersection of areas. A variety of other applications to analysis of boolean functions is given by (Michalski, 1978). In this paper we apply a recursive unfolding technique to visualising the surfaces relevant to evolutionary computation. Three case studies are described focusing on functions that have particular interest for researchers interested in the differences between mutation and crossover based search.

One of the early tenets of evolutionary computation was the "building block hypothesis" (Holland, 1975). This hypothesis proposed that a characteristic of many real-world problems is that solutions may be constructed from progressively higher-level combinations of building blocks. These blocks must be easy to identify (once discovered) and easily recombined into a wide variety of structures. Holland (2000) reasons that, "once a computer scientist starts thinking about building blocks as a source of innovation, the next obvious step is to look for algorithms that can discover and exploit building blocks" (p374).

Holland argues that many problems exhibit hierarchical structure and prescribes that optimisation algorithms should behave accordingly. His claim is that genetic algorithms do so because of the crossover operator of reproduction which allows the recombination of disparate building blocks. The performance of genetic algorithms on such hierarchically structured cost surfaces has been studied using a variety of test functions. These tasks include the Royal Road problems (Forrest and Mitchell, 1993; Mitchell et al., 1994). The original Royal Road problem was proposed to study recombination using crossover, but in search of clarity, it had only one fitness peak. It was discovered that hill-climbing strategies worked very well, better in fact than crossover (see Mitchell, 1996 for a good summary of the reasons).

Goldberg, Korb and Deb (1989) argued that to fully examine the utility of crossover, it is necessary to use deceptive functions in which the fitness landscape includes local optima that interfere with hill-climbing techniques. More recently, Watson and Pollack (1999) reviewed the tasks that have been used to study hierarchical structure across a variety of studies from the literature. They concluded that many of the tasks used to investigate such problems do not have the requisite structure to fully investigate the power of evolutionary algorithms using crossover. Their own example, H-IFF (hierarchical if-and-only-if; Watson et al., 1998) has the interesting property of symmetry around diametrically opposed fitness peaks, with many suboptimal peaks and consequently many local optima. The H-IFF function is designed to exhibit *hierarchical* modularity, particularly *recursive* modularity. (In hierarchical modularity higher-level modules are comprised of lower-level modules, in recursive modularity the fitness function of the higher-level modules is also of the same type as that of the lower-level modules.)

Evolutionary algorithms using recombination should produce scalable solutions to

complex design problems if the complexity is due to hierarchical structure. However, to search multiple levels of hierarchical structures, algorithms must maintain diversity of modules, yet continue to exert selection pressure to improve solutions: the classic exploration/exploitation dilemma of all search algorithms. For any given population size, as the number of levels and size of the problem space increase, a complexity crisis will arise when there is insufficient capacity in the population to achieve both goals (Kauffman, 1993). The optimal trade-off between exploration and exploitation is dependent on the particular task. Consequently, knowledge of the cost surface should guide selection of these meta-parameters, bringing us back to our original dilemma of deducing how the parameters should be set for a given cost surface.

This paper considers a visualisation technique that is applicable to cost-surfaces that are defined over moderate-dimensional binary spaces ($\{0,1\}^n$ for $2 < n < 16$) and is ideally suited to those which have a symmetric, multi-modal structure. It is thus also suitable for *hierarchically* and *recursively* structured functions as these exhibit multi-modal structure at multiple levels. This visualisation technique exploits the fact that recursive patterns in low dimensions indicate (symmetric) multi-modal structure in higher dimensions. Although humans do not readily visualise high-dimensional structures, we do readily perceive symmetries. Thus, insights into the higher-dimensional structure of the problem can be obtained from a lower-dimensional representation.

The features of these functions that we would most like to understand concern

- the distribution of optima
- the size and shape of basins of attraction around each local optimum
- the paths of increasing fitness on the surface.

It is possible to use statistical techniques to estimate the properties of such features for some problems (see Kauffman, 1993 for application to NK problems), but they require detailed mathematical insight and lack the efficacy of direct visualisation.

By contrast, the unfolding visualisation technique that we use is aimed at providing a readily understood description of a cost surface, providing researchers with direct insight into a problem. Consequently, we provide analytical case-studies with the technique for three different test functions that have previously been used as benchmark tests for evolutionary computation: Royal Road (Forrest and Mitchell, 1993), H-IFF (Watson et al., 1998), and Hierarchically Decomposable Functions (HDF; Pelikan and Goldberg, 2000)[1]

1.1 A Note on Terminology

Optimisation typically involves searching for the extremum of an objective function. Depending on the domain or field, this extremum may be either a maximum or minimum point of the function. Associated with this difference in optimisation goal is the range of nomenclature used to refer to the objective function (compare 'fitness function' with 'cost surface'). As the functions that we investigate in this paper have come from evolutionary computation we shall adopt this field's terminology: optimisation searches for the maximum of a fitness function (or fitness landscape).

2 Methods for representing cost surfaces: Hypercubes and hypergraphs

Hypergraphs are a technique for recursively representing all points in a boolean space on a two dimensional display. They are particularly applicable to multi-modal func-

[1] Not to be confused with Holland's hyperplane-defined functions (hdf; Holland, 2000).

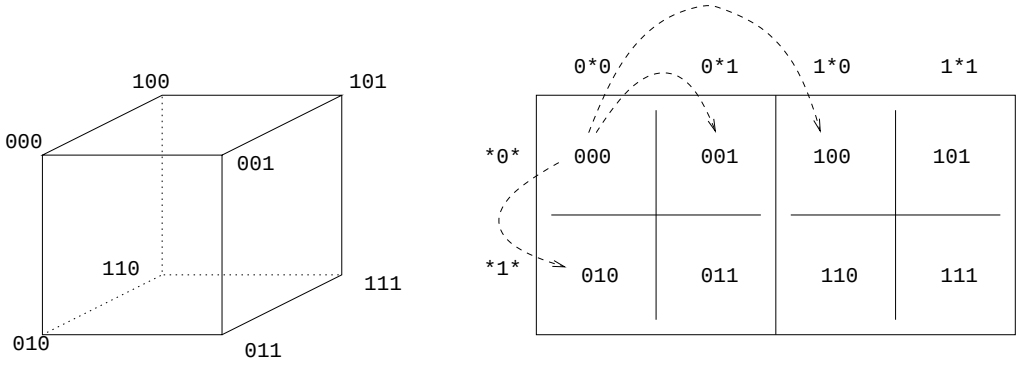


Figure 1: Hypercube and hypergraph. This figure demonstrates the relationship between the cube (left) and its graph form (right), showing the position in the graph to which each point in the cube is mapped. Rows and columns of the graph are labelled by their hyperplane templates. Dotted arrows on the graph show the immediate neighbours of 000.

tions. In this section we show how the corners of the $n-$dimensional hypercube (i.e., the points in the space) are mapped onto the two dimensional graph, and some of the insights that can be gained.

Consider an $n-$dimensional binary space, $\{0,1\}^n$. Each point in this space is an $n-$dimensional binary vector, $V = (x_0, x_1, \ldots, x_{n-1})$. The set of all possible points is the set of all possible bit-strings, $000\ldots0$ to $111\ldots1$. These strings can be represented as the corners of a hypercube. Higher-order hypercubes are recursively constructed from lower-order ones. A three-dimensional space has eight points which can be represented as the corners of a cube. The cube, and its unfolded hypergraph, are shown in Fig. 1. In the general case, an $n-$dimensional space has 2^n points. The corresponding hypergraph is a display of $2^{\lfloor n/2 \rfloor} \times 2^{\lceil n/2 \rceil}$ boxes, with each box corresponding to one corner of the hypercube. A sketch of the recursive structure for $n = 8$ is shown in Fig. 2.

Formally, using the recursive layout described above, for each $n-$bit string, $B = b_{n-1}b_{n-2}\ldots b_0$, the Cartesian co-ordinates (in binary notation) are given by $x = b_{n-2}b_{n-4}\ldots b_0$ and $y = b_{n-1}b_{n-3}\ldots b_1$. Thus, the lower order bits define the fine structure of the hypergraph and the higher order ones define the gross structure. The string of all zeroes ($000\ldots0$) maps to the top left corner, and the vector of all ones ($111\ldots1$) maps to the bottom right corner.[2]

Each point in the space has n immediate neighbours, which are the bit strings at a Hamming distance of one (or alternatively, single-bit mutations). In the graph, these neighbours are located at points $1, 2, 4, \ldots, 2^{n/4}$ positions away in the same row and column. It is often helpful to overlay gridlines on the graph which vary in thickness and highlight the recursive symmetries in the graph.

The regularity of the hypercube connection structure makes it possible to view the hypergraph without explicitly representing the neighbourhood topology. Since the hypergraph is a recursive unfolding of the hypercube, rather than a low-dimensional projection, each corner of the hypercube has a unique position on the graph and no information is lost in the process. Thus, if required, the positions of all connections can be

[2]The structure of the display can be tuned to the problem under investigation. For example, to keep dimensions that are adjacent in the vector representation together in the display, an alternative layout is to represent the lower order bits on the $x-$axis, and the higher order bits on the $y-$axis.

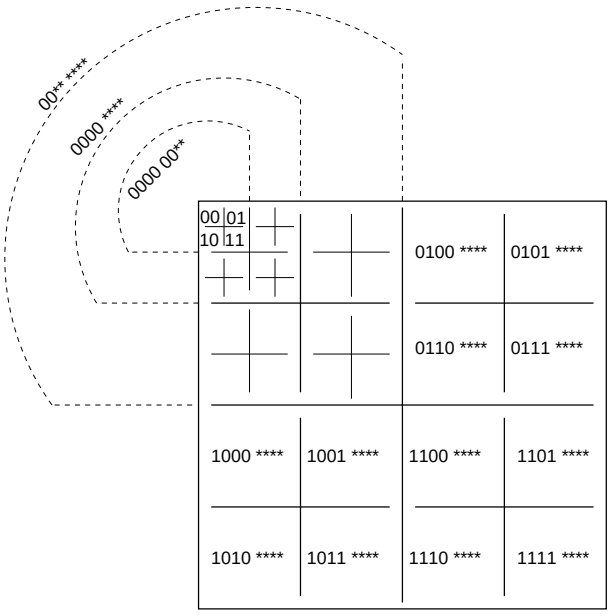


Figure 2: Hypergraph layout for an 8-dimensional space. The 8D hypercube is recursively unfolded to show all 256 strings, with 00000000 in the top left corner, and 11111111 in the bottom right corner.

inferred from the position of each string in the hypergraph. Note that the planar topology of the graph is *not* a straightforward representation of the high-dimensional space. Points that are adjacent to one another on the graph are not necessarily adjacent in the high-dimensional space. The important aspect to focus on is the recursive patterning of the graph.

3 Case Studies

In this section we present case studies of the visualisation approach. The visualisation approach has been implemented as a Java application.[3] The tool allows users to interactively explore the fitness landscape and can highlight a variety of features including the local optima, the basin of attraction of a given point (the set of points that can reach the target via neutral or positive mutations), the basin of potential of a given point (the set of points reachable by neutral or positive mutations), a random adaptive walk from a given point, and the local optima that can be reached by an adaptive walk from a given point. Two cases can be considered for random adaptive walks: a steepest-ascent walk which progresses via single-bit mutations that maximise fitness (locally), and a random-ascent walk which progresses via arbitrarily chosen fitness-enhancing single-bit mutations until a local optimum is reached.

With this tool, we explored three fitness function families. Each of these three functions has been used in the literature to examine crossover-based evolutionary search. Consequently these functions have a common structural property: they all demonstrate hierarchical modularity. In effect, hierarchical modularity is a direct instantiation of the

[3]The tool is available on the web at `http://www.itee.uq.edu.au/~btonkes/hsgp.html`. Source code is also provided.

Table 1: Schemata in 8-dimensional Royal Road and their respective fitness contributions. In this instance of Royal Road, all levels of the schema hierarchy contribute towards fitness (no missing levels).

Elementary schemata $u(e) = 1$ $\{e_i, i = 1, 2, \ldots, 8\}$	Two-combinants $u(c_2) = 2$ $\{e_i e_{i+1}, i = 1, 3, \ldots, 7\}$	Four-combinants $u(c_4) = 4$ $\{e_i e_{i+1} e_{i+2} e_{i+3}, i = 1, 4\}$	Eight-combinants $u(c_8) = 8$ $\{e_0 \ldots e_7\}$
1*******	11******	1111****	11111111
*1******	**11****	****1111	-
1***	****11**	-	-
1*	******11	-	-
****1***	-	-	-
*****1**	-	-	-
******1*	-	-	-
*******1	-	-	-

search function properties assumed by the building-block hypothesis. The basis of all of these functions is that fitness-contributing schema are composed of smaller fitness-contributing schema.

3.1 Case Study 1: Royal Road

The Royal Road (RR) class of functions was one of the earlier attempts at characterising the class of functions for which genetic algorithms are maximally suited (Mitchell et al., 1994). RR is defined over bit strings of length 2^l and has recursively defined schemata (see Table 1). In its original conception, 'levels' in the schema hierarchy of RR could be removed, thus increasing the size of neutral layers or 'plateaus' in the landscape. Here we consider the function having all possible $(l + 1)$ levels of schemata to show the maximum gradation in the graph and to highlight the recursive pattern in the graph for introductory purposes. Were levels to be removed from the function, the graph would show neutral layers as neighbourhood regions with identical greyscales. (Such neutral layers can be seen in other functions that have been implemented in the tool, for example, neutral variants of the NK landscape.)

We present an analysis of RR here not because it is particularly interesting or because visualisation provides novel insight into the function, but rather because RR is an intuitively easy to understand function and is helpful in introducing the hypergraph approach.

The hypergraph for eight dimensional ($l = 3$) RR is shown in Figure 3. The entire space comprises 256 (i.e, 2^8) bit-strings and all are shown on the hypergraph on a 16×16 ($2^4 \times 2^4$) grid. The recursive, and hence multi-modal, nature of RR is reflected in the recursive pattern of the graph. At the smallest scale, the graph has a lighter colours in the top left than in the lower right (with symmetry on the other diagonal). At the next higher level of scale (for example, in the top-right 1/16) this pattern repeats, as it does at larger scales. At this dimensionality (four levels of schemata hierarchy) it is straightforward to generalise the structure of the graph for higher dimensions. This observation makes one of the advantages of the hypergraph approach evident: the dimensional scaling properties of a function, which are often difficult to conceptualise, are made salient by two-dimensional pattern recognition.

Note that the smooth nature of RR appears as a non-monotonic pattern unlike a naive interpretation may suggest. However, a monotonic pattern can be observed by a *recursive* reading of the graph. As users become more experienced with the layout,

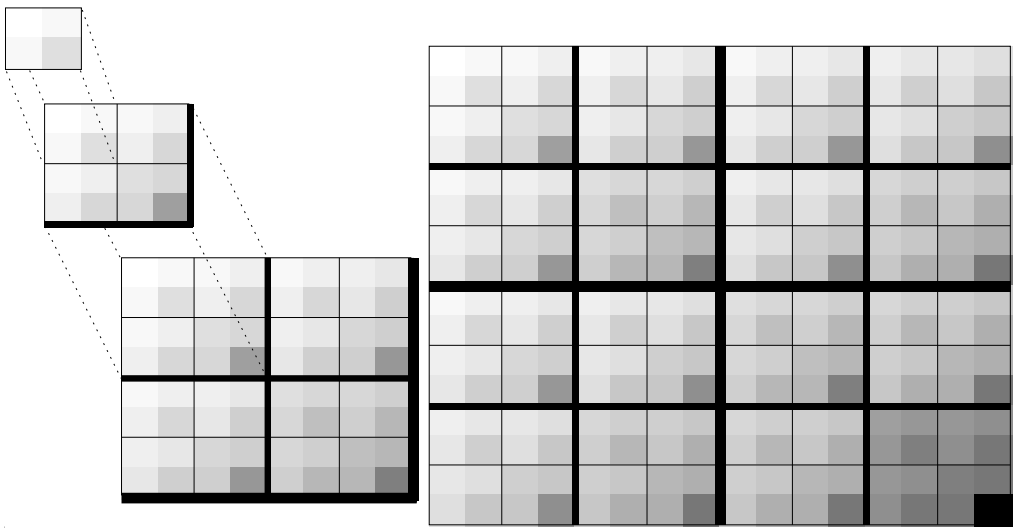


Figure 3: Hypergraph representation of eight dimensional Royal Road. The recursive layout is read by focusing first at a local scale, then at progressively larger scales, as shown to the left. Darker shaded areas correspond to points of higher fitness, and lighter shaded areas to points of lesser fitness. The global optimum of the solution, 11111111, lies in the bottom right corner of the graph. Note the recursive patterning of the graph indicating multi-modality.

this underlying recursive pattern can be recognised in many different functions. The benefits of mastering the recursive layout become apparent when considering more complex functions.

The simplicity of the structural properties of RR are mirrored in the simplicity of the graph structure. The function has a single global optimum (the string of all ones) and no local optima since changing any '0' to a '1' results in an increase in fitness. The recursive patterning of the graph, highlighted in Figure 3 indicates symmetric multi-modality. The low-order bits are represented in the graph as fine structure, while the high-order bits reveal themselves in the coarse structure. When the graph shows the same pattern at multiple scales it thus indicates symmetric modules. The size of these modules can be determined by the rate at which the graph pattern repeats, in the case of RR the pattern repeats over a 2×2 square indicating modules of 2 bits.

More than simple multi-modality, RR has a recursive structure. A recursive structure implies a non-linear interaction between the basic modules (i.e., the fitness benefit of a composite module is greater than the sum of fitness benefits for its elementary module components). It takes some familiarity with the hypergraph approach to recognise this non-linear interaction, but it can be readily observed in Figure 3 by noting that the global optimum at 11111111 seems substantially darker than its neighbours: it is clear that there is not a linear trend in increasing darkness.

Number of local optima. There is a single local optimum which is thus also the global optimum. The single optimum can be directly observed by a recursive reading of the hypergraph, but it can also be directly computed and highlighted by the tool (the tool provides the ability to colour those points that are local optima, or those that are global optima, etc.).

Size and shape of basins of attraction. As there is only a single local (global) optimum, the basin of attraction for this point is the entire space. This property is shown trivially with the tool by choosing to highlight the basin of attraction of the global optimum; the entire space becomes highlighted.

Average length of random adaptive walk. The length of random adaptive walks for RR defined on n dimensions scales as $O(n)$. The tool can show a random adaptive walk from a given point. Selecting the same point multiple times shows a variety of adaptive walks from that point. Consistent with an understanding of RR, the length of walks (the number of points highlighted) decrease linearly with the fitness value (grey value).

Given these insights into the structure RR derived from the hypergraph, it is unsurprising that a random-mutation hill-climbing algorithm can easily and quickly find the global optimum (Forrest and Mitchell, 1993).

3.2 Case Study 2: Hierarchical If-and-Only-If

The hierarchical-if-and-only-if (H-IFF) function is an extension to RR that adds many local optima to the landscape. For every schema in RR, H-IFF adds a complementary schema — the bitwise negation. The simplest way to define H-IFF for a bit-string x is $H(x) = RR(x) + RR(\overline{x})$, where $\overline{x}$ is the bitwise negation of x. As with RR, H-IFF can be considered with a restricted number of levels in the schema hierarchy (Wiles et al., 2001), but here we use all levels of the hierarchy. Although H-IFF is simply the summation of two fitness functions with simple structural properties, the result is a fitness landscape of substantially greater complexity.

In the original paper that introduced the H-IFF function, an 'artist's impression' of the landscape was provided in the form of a tour through the landscape from one peak to the other, traversing the local optima. This one dimensional representation provides a good intuition with respect to landscape structure but is necessarily a limited subset of the space, and can be misleading with respect to the relationship between high and low points in the space. The visualisation technique provides a complete, unambiguous representation of the landscape and shows properties that cannot be derived from the line graph.

The greater complexity of H-IFF better demonstrates the features of the hypergraph visualisation technique. Consideration of a simple instantiation of H-IFF, defined over only four dimensions is instructive as to how to navigate the hypergraph (see Figure 4).

Exploring the hypergraph for a larger space (see Figure 5) shows the generality of the structures observed in Figure 4.

Number of local optima. The local optima of the function are all of the points along the diagonal (as observed in the four-dimensional case). As the dimensionality of the problem increases, this topological relationship is consistent. It therefore provides a direct intuition that the number of local optima scales with the width and height of the graph: as $2^{n/2}$ for a function defined on n dimensions. This relationship can be easily verified from an analysis of the H-IFF function. The value of the graphical representation is that it provides immediate feedback about which properties of a function are of particular interest. Furthermore, there are an equal number of local minima that fall along the other diagonal, all of which are of equal, minimal fitness.

Size and shape of basins of attraction. Some insights can be drawn about the size and shape of the basins of attraction for the local optima, shown in Figure 6. The most obvious observation is that the basin of attraction for the global optima is a Sier-

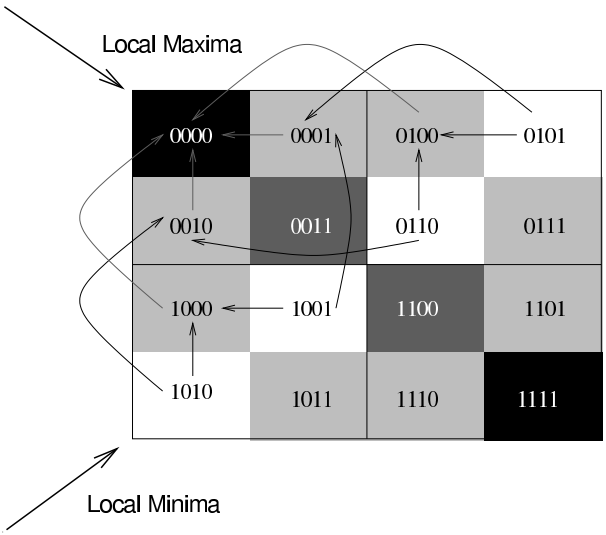


Figure 4: Hypergraph representation for H-IFF defined on four dimensions. Using the interactive hypergraph tool reveals that there are four local optima, all of which fall along the top-left/bottom-right diagonal, and that the points of least fitness fall along the other diagonal. Also shown in this figure are all of the paths of increasing fitness that lead to one of the global optima (0000). The set of points through which these paths travel form the basin of attraction for 0000. The basin of attraction for the other global optimum can be easily derived through the symmetry of the graph.

pinski triangle,[4] made more obvious by the manner in which the points have been highlighted. The recursive pattern reveals that the size of each basin is given by $3^{n/2}$ (3 corners of each square with $n/2$ levels of recursive triangles). Thus, the basins scale in proportion to *(the number of points in the Sierpinski triangle) / (the size of the space)*, or $3^{n/2}/2^n = (3/4)^{n/2}$. While the formal mathematical analysis is tractable for this function, the visualisation was the stimulus for conceptualising the basin sizes in terms of their fractal structure.

Not only does the basin of attraction for the global optima form a Sierpinski triangle, the basins of attraction for all of the local optima are also variants of the Sierpinski triangle (see Figure 6(b)). Thus, a cursory visual inspection using the hypergraph tool reveals that the basins of attraction for all local optima are of the same size. The size, shape, and overlap of these basins suggest that random adaptive walks are as likely to terminate at a global optimum as they are to terminate at a local optimum (i.e., all outcomes are equally likely). Further detailed analysis revealed that this speculation was indeed correct.

Interactions between basins of attraction. Such a landscape is very difficult to optimise using hill-climbing approaches. The only points in common between the basins of attraction for 00000000 and 1111111 are the points of least fitness which lie along the diagonal. In fact, these points belong to the basin of attraction for all local optima, or conversely, any local maximum can be reached by an adaptive walk from these minima. The separation of the basins at such a low level of fitness indicates a watershed in

[4]The Sierpinksi triangle is a well-known fractal forming a triangle in which the centre is recursively removed.

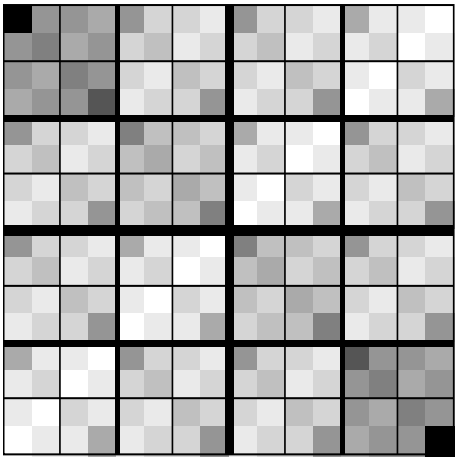


Figure 5: Eight dimensional H-IFF. As in Figure 4, all local optima fall along the top-left/bottom-right diagonal. Compare the structure of H-IFF with that of RR, shown in Figure 3. The landscape of H-IFF is clearly more complex than that of RR which has only a single fitness peak and which is, effectively, a smooth slope.

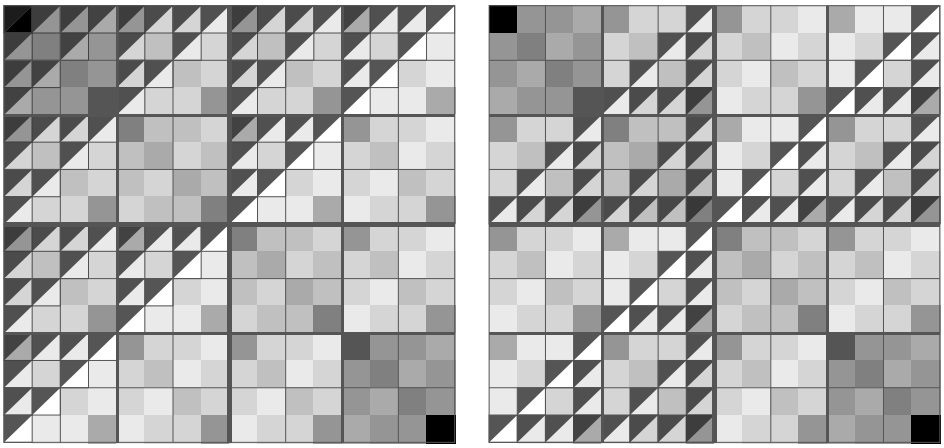
the landscape that would be difficult to traverse with hill-climbing algorithms. The H-IFF task was designed to be difficult for mutation alone, and the hypergraph makes the success of this design goal directly apparent. A benefit of the visualisation approach is to confirm intuitions about landscape properties such as the watershed between global optima.

Average length of a random adaptive walk. Initial exploration with the tool suggested that for H-IFF there was no difference between steepest-ascent walks and random-ascent walks. For many points this assertion is true: all fitness-improving mutations are equally beneficial. Further exploration shows that for points that are neighbours of local optima, a difference occurs. A simple calculation reveals that the n local optima have $n^2/2$ unique neighbours in a space of 2^n points. As the size of H-IFF increases $n^2/2$ becomes vanishingly small compared with 2^n so the benefits of steepest-ascent over random-ascent should be expected to diminish.

Another feature that becomes apparent through exploration with the tool is that all adaptive walks from any particular point are of equal length (and steepest-ascent walks are of equal length to random-ascent walks). As for RR, the length of these walks on n dimensions scales as $O(n)$.

3.3 Case Study 3: Hierarchically Decomposable Functions

Unlike RR and H-IFF, hierarchically decomposable functions (HDFs) were designed as a class of functions for which standard GAs would *not* be suitable (Pelikan and Goldberg, 2000). HDFs can be viewed as a generalised form of H-IFF. The major structural difference between HDFs and H-IFF (and RR) is that HDFs need not possess 'hierarchical consistency'. In H-IFF the fitness contribution from a building block at a particular level is indicative of its utility as a building block at the next higher level of the hierarchy. In HDFs the function for determining fitness contribution can be modified at each level of the hierarchy. Whereas H-IFF has recursive modularity, HDFs assume only hierarchical modularity.



(a) (b)

Figure 6: Basins of attraction for eight-dimensional H-IFF. Shown are (a) the basin of attraction for the global optimum, 00000000 and (b) the basin of attraction for the local optimum 00111111 (the bottom right point in the upper left quadrant). Points in the basin have been highlighted by drawing a triangle across the upper left half of the area. (In the interactive tool the whole square is coloured; this alternative representation is more appropriate for greyscale media.)

Although HDFs define a general class of function, in this section we shall focus on one example, a simplified variant of a function studied by Pelikan and Goldberg (2000) . In its original form, the function is a hierarchical, multi-modal, bipolar, deceptive function (a non-hierarchical version had been previously studied (Goldberg et al., 1992) and has elsewhere been termed M7 (Mahfoud, 1995)). The size of this problem scales as 6^n, so showing just two levels of hierarchy would require 36 dimensions (2^{36} points) — far beyond the capacity of the hypergraph tool (or any approach to visualise the complete landscape). By removing some symmetries in the function (removing bipolarity) we can make the function scale as 4^n; small enough to be able to visualise two levels of hierarchy (non-hierarchical variants of this form have also been previously studied (Liepins and Vose, 1990)). The simplified function still contains important structural properties: hierarchy, multi-modality, and deceptiveness.

The simplified function, henceforth *sHDF*, is defined on non-overlapping modules of length four. Defined on 16 dimensions, there are thus four elementary schemata regions. The fitness contributions of elementary schemata are a function of the number of ones (unitation) in the appropriate region of the solution vector, u, and is given by

$$f(u) = \begin{cases} 0.9 \text{ if } u = 0 \\ 0.8 \text{ if } u = 1 \\ 0.5 \text{ if } u = 2 \\ 0.0 \text{ if } u = 3 \\ 1.0 \text{ if } u = 4 \end{cases} \quad (1)$$

Thus, schemata $1001{*}{*}{*}\ldots*$ and ${*}{*}{*}{*}0110{*}{*}\ldots*$ both have fitness contributions of 0.5. The next level of the hierarchy is computed similarly, with the difference being that

Table 2: Evaluation of sHDF for a sample solution, 0110 0111 1100 1111.

Bit string	**0**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**
Value of elementary modules, based on unitation: $f(u)$.	0.5				0.0				0.5				1.0			
Hierarchic interpretation of module: 1 if $u > 2$ 0 otherwise.	**0**				**1**				**0**				**1**			
Fitness contribution of higher-level module (0101).	0.5α															

an entire module must be tagged as being either a one or a zero (since f only considers strings of length four). A module is considered a one if three or more of its bits are one, and as a zero otherwise. The evaluation for a sample solution is given in Table 2. Given the bitstring 0110 0111 1100 1111, the elementary modules provide fitness contributions of 0.5, 0.0, 0.5, and 1.0, based on their unitation. For computing the fitness contribution of the next level of the hierarchy, this vector is recoded as 0101 which has a fitness contribution of 0.5. For this higher level of the hierarchy, the fitness may be scaled by a factor, α (by default 1). The fitness of the entire string is thus $2.0 + 0.5\alpha = 2.5$.

Although we have reduced the size of the problem from 32 to 16 dimensions, which can be effectively displayed on a high-resolution display, this graph does not translate well to paper (see Tonkes, 2002 for the full display). Consequently we are able to show only a relatively small portion of the graph (see Figure 7) covering three elementary schemata regions (12 dimensions). Figure 7 clearly shows the deceptive nature of sHDF. This graph shows that the deceptive optimum, shown in the top left, sits atop a large, smooth slope, not unlike RR (but in the opposite direction in this case). Conversely, the paths leading to the global optimum comprise only a small fraction of the space.

For a user experienced with the hypergraph visualisation technique, the starting point for interpreting this graph is the lowest level of recursive structure, shown in Figure 8. What we see in this graph is that there are two local optima, in the top left and bottom right positions in the graph, and that for the optimum in the lower right its four immediate neighbours are all white, showing minimal fitness. In contrast, the top left optimum's immediate neighbours are of almost equal fitness so that the 'downhill' slope from this optimum is much larger than that from the bottom right optimum. This recursive structure is reflected in the larger-scaled structures.

Note that the hypergraph of sHDF has a highly recursive structure but that this structure is *not* due to the hierarchical properties of the function. Rather, it is a consequence of the repeated, low-level modular structure. Schemata that cover low-order bits appear in the graph as high-frequency patterns. Schemata of higher-order bits appear as low-frequency patterns. Figure 7 shows three 'modules' and there are consequently three levels of recursive structure in the graph. The basic unit of repetition comprises sixteen points, the number of points in each module, and is shown in Figure 8.

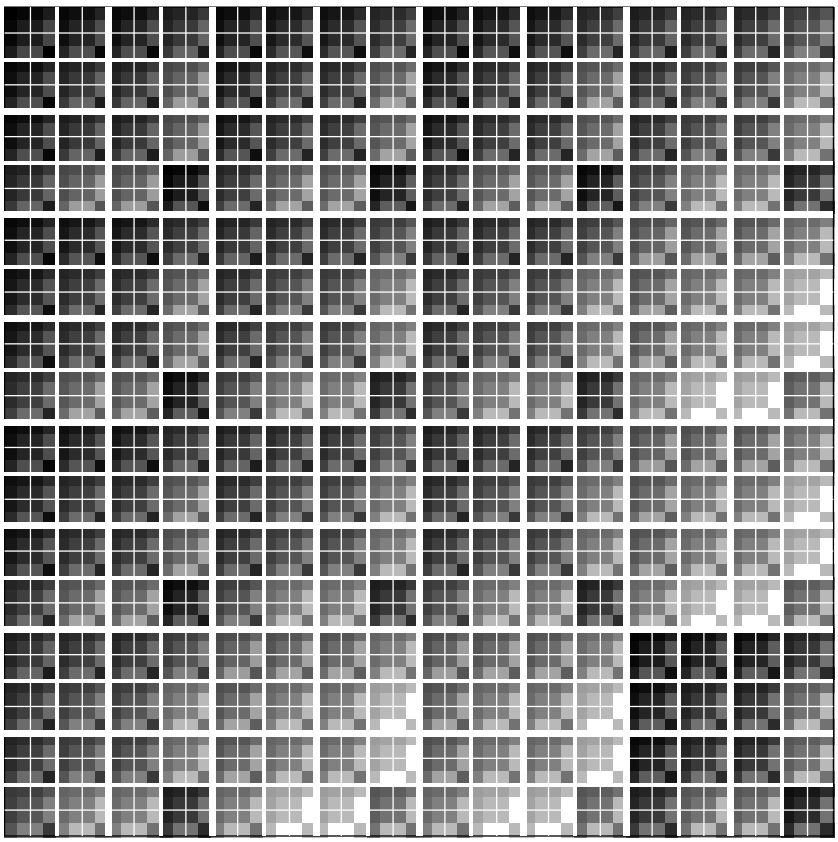


Figure 7: A portion of the landscape for the simplified hierarchically decomposable function, sHDF. The portion shown corresponds to the top-left 1/16th of the full graph, or, that part of the graph corresponding to the schema 0000 **** **** ****. Although the complete graph cannot be shown, the remainder can be recursively extrapolated from the area shown.

Surprisingly, the addition of the fitness contributions from the next higher level in the hierarchy has relatively little impact on the general structure of the fitness landscape. That is, there is little change in the landscape when varying the scaling factor, α, from 0 to 4. We consider analysis of the fitness landscape for both $\alpha = 0$ and $\alpha = 1$ (there is effectively no difference between $\alpha = 1$ and $\alpha > 1$). For $\alpha = 0$ the function is no longer hierarchical, but remains multi-modal. The general hypergraph structure for $\alpha = 0$ can be recursively extrapolated from Figure 8 (see also Figure 7) since no high-order changes are made other than the continuation from the recursive pattern. The case for $\alpha = 1$ will be described in terms of how it differs from the baseline ($\alpha = 0$) case.

Number of local optima. As for H-IFF, using the tool to highlight the local optima shows a recursive distribution. The basic unit of recursion (Figure 8) has two local optima (0000 and 1111). As the number of dimensions under consideration increases, the local optima scale recursively into diagonally-opposite corners. Thus, as each additional module is considered, the number of local optima doubles, so that for the full

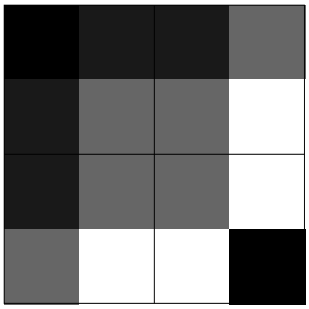
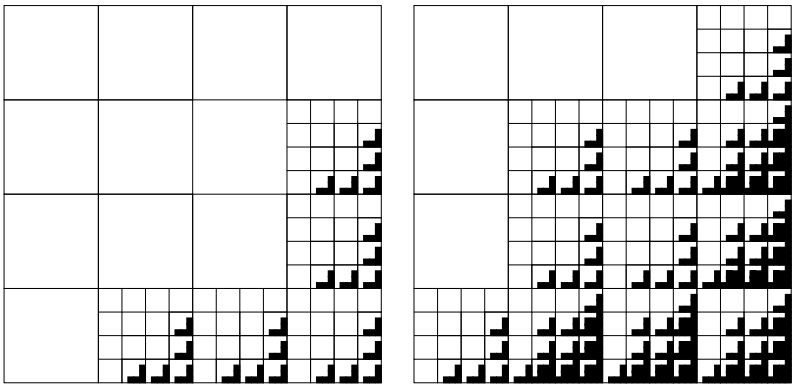


Figure 8: One 'module' of sHDF, defined over 4 bits. This fragment of the complete graph is the schema 0000 0000 0000 ****.



(a) $\alpha = 0$ (b) $\alpha = 1$

Figure 9: Schematic showing the lower-right portion of the basin of attraction for global optimum of (a) non-hierarchical and (b) hierarchical forms of sHDF. The section of the complete graph shown in this figure is the schema 1111 **** **** ****. For the full graph, an extra level of recursive patterning needs to be added.

16 dimensions of sHDF there are $2^4 = 16$ local optima. Furthermore the points that are the local optima for $\alpha = 0$ are also the local optima for $\alpha = 1$: adding the hierarchical fitness contribution does not alter the number or position of the local optima.

Size and shape of basins of attraction. Unlike RR and H-IFF, the local optima of sHDF have basins of attraction of widely varying sizes. The tool makes this difference in size immediately obvious. Two cases are of particular interest: the global optimum at $11\ldots1$ has a very small basin of attraction and the 'deceptive' local optimum at $00\ldots0$ has a very large basin.

The tool makes it readily apparent that in contrast to the local optima, there is a difference in the basins of attraction for the cases of $\alpha = 0$ and $\alpha = 1$. For $\alpha = 0$, the case is relatively straightforward. Consider the local optima in the module of Figure 8. The deceptive optimum has a basin of attraction containing all points except the global optimum. This pattern is repeated recursively in the hypergraph with the basin for the deceptive function comprising $15/16$ of the space at each recursive level.

The proportion of the entire space in the basin of attraction is thus $(15/16)^4$. The basin for the global optimum in Figure 8 comprises five points: the optimum itself and its immediate neighbours (the lightest shaded points in the figure). Again this pattern is recursive, so that the proportion of the space in the basin of attraction for the global optimum is $(5/16)^4$. The shape of the basin is shown schematically in Figure 9.

The situation for $\alpha = 1$ is more complex. While the basins of attraction for both the deceptive and global optima are broadly similar to their $\alpha = 0$ counterparts, there are some interesting differences. With the addition of the hierarchical fitness contribution, points such as 0011 1111 1111 1111 fall into the basin of attraction of the global optimum (since changing the first bit would lead to an improvement in fitness). Indeed, the basin of attraction for the global optimum becomes noticeably larger. This basin follows a recursive pattern, which scales as $(11+6(k-1)) \times 5^{k-1}$, where k is the number of levels of hierarchy in the recursive pattern of the graph (four in this case). In terms of k, the size of the space grows as 2^{4k}. Hence, as the function is expanded to more levels of hierarchy, the proportion of the space in the basin of attraction for the global optimum rapidly diminishes. The differences in the basins of attraction for the global optimum for $\alpha = 0$ and $\alpha = 1$ are apparent in Figure 9.

The addition of the hierarchical fitness component of sHDF makes little change to the basin of attraction for the deceptive optimum. Only a relatively small number of points close to the global optimum are lost from the basin. As the size of the function is increased, it should be expected that this basin of attraction remains largely conserved when adding the hierarchical component.

Interactions between basins of attraction. By comparing the basins of attraction for different optima (each can be alternately highlighted) it is obvious that in the simple case ($\alpha = 0$) the basins for the deceptive optimum and the global optimum have no points in common. For the $\alpha = 1$ case, there is minimal overlap, with only some local minima belonging to both basins.

As the dimensionality of sHDF increases, the proportion of the space in both the deceptive and global optimum approaches zero. That is, most of the space begins to fall into the basins of attraction of the increasingly numerous local optima. However, the basins of attraction for the deceptive optimum remain the largest of all basins.

4 Evaluation of Hypergraph Effectiveness

Hypergraphs provide a mechanism through which properties of high-dimensional surfaces can be explored with relative ease. Visualisation provides direct insight into landscape structure rather than requiring mathematical insight. Visualisation techniques such as the hypergraph approach should not be viewed as supplanting mathematical analysis, rather, they provide additional modality for reasoning about the structure of fitness landscapes and can be used for quickly identifying potentially interesting structural features.

Early unpublished experiments on the usability of hypergraphs showed that for six-dimensional hypercubes, people either mastered the display and learned to navigate very quickly — usually in time linearly proportional to the Hamming distance between points — or failed to understand the structure at all. One subject, a computer science graduate student, not only had the fastest times of all of the people tested, but also appeared to navigate between points in constant time. Clearly the technique requires considerable effort before the high-dimensional structure becomes intuitively navigable. For example, in the Royal Road diagrams, beginners are apt to interpret the point 00001111 (at location $(4, 4)$ in Fig 3) as a local optimum because it is a dark

point surrounded by lighter coloured points. After some experience with the tool, users appreciate that the planar topology of the graph is not in direct correspondence with the high-dimensional topology and that its neighbours are the dark points at $(8, 4)$ and $(8, 8)$.

The hypergraph visualisation technique is not without its shortcomings. Primary amongst its limitations is its lack of scalability. Because the hypergraph attempts to represent the entire surface (a worthy goal), the size and complexity of the display scales exponentially in the dimensionality of the fitness function. The approach is not aimed at use in exploring real-world problems, since if a fitness landscape can be enumerated, the function poses little practical difficulty. However, for artificially constructed test functions, the cost surfaces for problems of 'interesting' size are often too large to represent. For example, Holland's hyperplane defined functions (hdf) (Holland, 2000) are defined over spaces of hundreds of dimensions.

Nevertheless, some simplifications can be made that allow these larger, and more complex cost functions to be reduced to manageable sizes, as in the case of sHDF. As another example, much of the space in hdfs is a flat plateau of uniform, minimal fitness. These 'uninteresting' dimensions can be removed from the graph. On these simplified versions of hierarchically defined test functions, the hypergraph display provides useful intuitions of the high-dimensional structure. Furthermore, understanding how various landscape features come to be represented in the pattern of the graph can give important intuitions about how the graph would look if it could be plotted. For moderate sized problems ($16 \leq n \leq 32$), where a function is small enough to enumerate yet too large to easily graph, it would be possible to extend the tool to allow the display to be zoomed and panned. Alternatively, the information in the graph could be 'compressed' so that rather than each point in the graph representing a point in the space, a point in the graph could instead summarise the information of some schema (Mihalisin et al., 1991). For example, when reducing a graph for $n = 20$ to $n = 16$ the top left position of the graph could show the maximum, average or minimum fitness for all points in the schema 0000 0000 0000 0000 ****, and similarly for other entries in the graph.

The special case of functions with binary fitnesses, the visualisation technique can potentially scale to an arbitrarily large number of dimensions since the graph can be arbitrarily compressed without losing information. This special case has been used for visualising population distributions (Collins, 1997). It has also been suggested as a useful technique for visualising quantum computation (Lee Spector, *pers. comm.*).

A further issue with the hypergraph approach when considering functions of the type considered here is the mismatch between the topology of the graph and the manner in which GAs search. The unfolding of the space into the hypergraph structure is done so that neighbouring points are laid out in a recursive fashion. That is, the layout of the hypergraph is designed to visually emphasise the *Hamming distance* between points: translational adjacency in the hypergraph topology represents single-point mutation. The layout of the hypergraph is thus tuned to display how the space would be searched by a hill-climbing approach. Crossover works by combining subsequences from potentially disparate points. The hypergraph does not make it clear how a population of solutions would adapt when crossover is used, but this dilemma is common to all landscape techniques: the problem is one of the landscape metaphor.

5 Discussion

The three functions considered in this paper — RR, H-IFF, and HDF — were all proposed in the literature to explore the limits of crossover-based genetic search. RR was

a first (unsuccessful) attempt to characterise the simplest form of function for which crossover-based search would outperform mutation-based search. The hypergraph representation (Figure 3) reveals the simplicity of this function; which is, in effect, a smooth slope that mutation-based search should be expected to climb quickly.

H-IFF represents a later attempt to characterise a function which crossover-based search, but not mutation-based search, can efficiently optimise. The definition of H-IFF is little more complex than the definition of RR, yet the difference in structural complexity is massive (compare Figure 3 with Figure 5). The insights provided by the hypergraph visualisation make it clear why mutation-based search is successful on the RR landscape but not the H-IFF landscape.

Observations that for all local optima the basins of attraction are equally sized in H-IFF prompted the hypothesis that all outcomes of random-adaptive walks are equally likely. The graphs also made it clear how the number of local optima scale with the size of the problem. From these two properties it follows that the size of the population for mutation-based search must scale with the number of local optima ($2^{n/2}$). Thus, from visualisation of the cost surface we (a) gain insight into the most desirable parameters for mutation-based search, and (b) discover that module recombination is a more appropriate approach.

Comparing the visual properties of H-IFF to those of RR reveals an obvious difference. Relating the performances of different algorithms on these two cost surfaces to features in their visual appearance provides insight into novel problems. Upon recognising that the pattern of RR represents a smooth slope it becomes easy to identify this pattern as a feature of other surfaces (e.g., smooth NK functions). The efficacy of the visual display provides insights into appropriate similarity metrics between cost surfaces (i.e., the features that are important for a particular optimisation algorithm). For example, recursive structure in high dimensions produces repeated patterns in a low dimensional unfolded representation.

HDF attempts to characterise the capabilities of crossover-based search in a manner contrary to RR and H-IFF. Rather than attempting to demonstrate the 'lower limit' of GAs (the simplest functions for which crossover becomes beneficial), HDFs are aimed at being a class of function which are *not* amenable to search by standard crossover techniques. It is not entirely clear from the hypergraph of sHDF (Figure 7) why crossover should fail. What the graph does obviously show is the vast region of space that leads to the deceptive optimum at $000\ldots0$. The difficulty in the analysis stems from the representation of the landscape from a mutation perspective rather than a crossover perspective, one of the deficiencies of the hypergraph approach discussed earlier. Certainly an inspection of the graph would predict that mutation-based search should fail to find the global optimum (as it does) due to the presence of the deceptive 'slope'. It would seem likely that crossover-based approaches might also fail to avoid climbing the deceptive slope.

Other functions that are of interest to EC researchers are those with interesting epistatic interactions (i.e., interdependences between variables). The tool implements Kauffman's NK function and two of its neutral variants (NKp and NKq) as a paradigmatic example of tunable epistasis and the effects of neutrality. See (Tonkes, 2002) for further details.

6 Conclusions

The visualisation approach taken in this paper offers some benefits over the more traditional statistical and mathematical analysis of cost surfaces. One benefit is that it is

more accessible to those lacking a mathematical background. Another is that it provides an exploratory tool: interesting features 'pop out' and can then be subjected to a more rigorous analysis as was the case with H-IFF, where we determined that all local optima were equally likely to be found by a random adaptive walk.

The hypergraph visualisation technique was designed for exploring fitness landscapes that possessed a symmetric, multi-modal structure defined over a binary space. To this end, the technique is successful (modulo the reservations above) in that it uses the human perceptual system's ability to perceive patterns as a way to enhance the salient features of high-dimensional structure. We have also successfully used hypergraphs for investigating landscapes that are neither hierarchical nor multi-modal, such as NK functions (Kauffman, 1993), the energy function defined over the states of Hopfield networks and the changing probability distribution of population based incremental learning (PBIL) (Baluja, 1994) as it searches a fitness function.

The case of PBIL is particularly interesting as it affords a direct comparison between the model's probability distribution and the fitness landscape, and highlights the lack of modelling of the joint-probability distribution of the functions variables. In future work we would like to implement models such as the Bayesian Optimisation Algorithm (Pelikan and Goldberg, 2000) which do attempt to incorporate the joint probabilities so that we can visualise the correspondence between the landscape and the probability distribution. In this way we hope to better understand the interaction of this class of optimisation algorithms with their landscapes so that we might better understand how and when to apply a particular model.

Acknowledgements

This work was supported by an ARC grant to Janet Wiles.

References

Baluja, S. (1994). Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning,. Technical Report CMU-CS-94-163, Carnegie-Mellon University, Pittsburgh, PA 15213.

Belding, T. C. (2001). Potholes on the royal road. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 211–218, San Francisco, CA. Morgan Kaufmann.

Buja, A. and Asimov, D. (1986). Grand tour methods: An outline. In *Proceedings of the 18th Symposium on the Interface*, pages 63–67. American Statistical Association.

Christensen, S. and Oppacher, F. (2001). What can we learn from No Free Lunch? A first attempt to characterize the concept of a searchable function. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1219–1226, San Francisco, CA. Morgan Kaufmann.

Collins, T. D. (1997). Using software visualization technology to help evolutionary algorithm users validate their solutions. In Baeck, T., editor, *The Proceedings of The Seventh International Conference on Genetic Algorithms*, pages 307–314. Morgan Kaufmann.

Forrest, S. and Mitchell, M. (1993). What makes a problem hard for a genetic algorithm? Some anomalous results and their explanation. *Machine Learning*, 13(2/3):285–319.

Friedman, J. H. and Tukey, J. W. (1974). A projection pursuit algorithm for exploratory data analysis. *IEEE Transactions on Computers*, 23(9):881–889.

Goldberg, D. E., B.Korb, and Deb, K. (1989). Messy genetic algorithms: Motivation, analysis and first results. *Complex Systems*, 3:493–530.

Goldberg, D. E., Deb, K., and Horn, J. (1992). Massive multimodality, deception, and genetic algorithms. In Männer, R. and Manderick, B., editors, *Parallel Problem Solving from Nature, 2*, pages 37–46. Elsevier.

Hill, F. J. and Peterson, G. R. (1968). *Introduction to switching theory and logical design*. Wiley, New York.

Hinton, G. E., Sejnowski, T. J., and Ackley, D. H. (1984). Boltzmann machines: Constraint satisfaction networks that learn. Technical Report CMU-CS-84-119, Carnegie-Mellon University, Pittsburgh, PA 15213.

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

Holland, J. H. (2000). Building blocks, cohort genetic algorithms, and hyperplane-defined functions. *Evolutionary Computation*, 8(4):373–391.

Kauffman, S. (1993). *The Origins of Order*. Oxford University Press, NY.

LeBlanc, J., Ward, M., and Wittels, N. (1991). Exploring n-dimensional databases. In *Proceedings of the IEEE Conference on Visualization 1991*, pages 230–237.

Liepins, G. E. and Vose, M. (1990). Reresentational issues in genetic algorithms. *Journal of Experimental and Theoretical Artificial Intelligence*, 2:101–115.

Mahfoud, S. W. (1995). A comparison of parallel and sequential niching methods. In Eshelman, L., editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 136–143, San Francisco, CA. Morgan Kaufmann.

Michalski, R. S. (1978). A planar geometric model for representing multidimensional discrete spaces and multiple-valued logic functions. Technical Report UIUCDCS-R-78-897, University of Illinois at Urbana-Champagne.

Mihalisin, T., Timlin, J., and Schwegler, J. (1991). Visualizing multivariate functions, data, and distributions. *IEEE Computer Graphics and Applications*, pages 28–35.

Mitchell, M. (1996). *An introduction to genetic algorithms*. MIT Press, Cambridge, MA.

Mitchell, M., Holland, J., and Forrest, S. (1994). When will a genetic algorithm outperform hill climbing. In *Advances in Neural Information Processing Systems*, volume 6, pages 51–58.

Munro, P. W. (1992). Visualizations of 2-d hidden unit space. In *International Joint Conference on Neural Networks*, volume 3, pages 468–473. IEEE.

Pelikan, M. and Goldberg, D. E. (2000). Hierarchical problem solving by the bayesian optimization algorithm. IlliGAL Report No. 2000002, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL.

Pryke, A. N. (1996). *The Haiku Visualisation System*. PhD thesis, University of Birmingham.

Shine, W. and Eick, C. (1997). Visualizing the evolution of genetic algorithm search processes. In *Proceedings of the 1997 IEEE International Conference on Evolutionary Computation*, pages 367–372.

Tonkes, B. (2002). Hyperspace graph paper WWW page. http://www.itee.uq.eu.au/∼btonkes/hsgp.html.

Tufte, E. R. (1992). *The Visual Display of Quantitative Information*. Graphic Press, reprint edition edition.

Watson, R., Hornby, G., and Pollack, J. (1998). Modeling building-block interdependency. *Parallel Problem Solving from Nature, proceedings of the Fifth International Conference*, pages 97–106.

Watson, R. and Pollack, J. (1999). Hierarchically-consistent test problems for genetic algorithms. In Angeline, P. J., Michalewicz, Z., Schoenauer, M., Yao, X., and Zalzala, A., editors, *Proceedings of 1999 Congress on Evolutionary Computation*, pages 1406–1413. IEEE Press.

Wiles, J. and Bloesch, A. (1992). Operators and curried functions: Training and analysis of simple recurrent networks. In Moody, J. E., Hanson, S. J., and Lippmann, R. P., editors, *Advances in Neural Information Processing Systems 4*, pages 325–332, San Mateo: CA. Morgan Kaufmann.

Wiles, J., Tonkes, B., and Watson, J. R. (2001). How learning can guide evolution in hierarchical modular tasks. In Moore, J. D. and Stenning, K., editors, *Proceedings of the 23rd Annual Conference of the Cognitive Science Society*, pages 1130–1135.

Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82.