Embedded Landscapes

Robert B. Heckendorn heckendo@cs.uidaho.edu
Department of Computer Science, University of Idaho, Moscow, ID 83844-1010, USA

Abstract

In this paper we introduce embedded landscapes as an extension of NK landscapes and MAXSAT problems. This extension is valid for problems where the representation can be expressed as a simple sum of subfunctions over subsets of the representation domain. This encompasses many additive constraint problems and problems expressed as the interaction of subcomponents, where the critical features of the subcomponents are represented by subsets of bits in the domain. We show that embedded landscapes of fixed maximum epistasis K are exponentially sparse in epistatic space with respect to all possible functions. We show we can compute many important statistical features of these functions in polynomial time including all the epistatic interactions and the statistical moments of hyperplanes about the function mean and hyperplane mean. We also show that embedded landscapes of even small fixed K can be NP-complete. We can conclude that knowing the epistasis and many of the hyperplane statistics is not enough to solve the exponentially difficult part of these general problems and that the difficulty of the problem lies not in the epistasis itself but in the interaction of the epistatic parts.

1 Introduction

Two classic and important problems have surprisingly similar mathematical structure: kSAT problems and NK-landscapes. These are examples of a larger and more general class of functions in which many small segments of a problem representation interact in intricate ways to yield a more complex function. Even though these functions are complex, their basic structure is decomposable and their epistasis is limited. It is the interactions of these simple parts that make the function hard.

In this paper we explore the idea of functions of limited epistasis and the mathematical model of such functions called embedded landscapes. We discuss how deeply constrained this class of functions is. We show how to compute the degree of bit interactions in their domain and where those interactions are absent. We show that because the underlying structure is simple, we have leverage enough to compute in polynomial time many statistics about the function values in hyperplanes that might control the trajectory of an evolutionary algorithm. However, even with the ability to compute all this information in polynomial time, these supposedly simple, highly constrained functions remain NP-complete.

2 Notation

We begin by introducing our basic notation and a useful numbering scheme for hyperplanes.

2.1 Some Simple Bit Operations

- The space of N bit strings is denoted $\mathcal{B}^N$.

- $[i]$ denotes extracting the i^{th} bit. So if bits are numbered from the right beginning with position 0, then $x =$`1111011` and then $x[2] = 0$.
- $\vec{1}$ will denote a string of all `1`'s. The length will be clear by the context of use.
- $i \subseteq j$, where $i, j \in \mathcal{B}^L$ reads as "i is contained in j" and j is said to "contain" i. That is, wherever there is a `1` in position i there is a `1` in j.
- The bit count or unitation function *bc*(i) returns the number of `1`'s in i. For example *bc*(`001011`) $= 3$ and *bc*$(15) = 4$.

Another important concept is the packing and unpacking of bitstrings by a mask. A function *pack* is defined as *pack* : $\mathcal{B}^L \times \mathcal{B}^L \rightarrow \mathcal{B}^M$, where $M \leq L$. *pack*(x, m) takes the bits in x and masks them with an L bit mask m such that *bc*$(m) = M$. The bits selected by the mask are then right justified with zero fill in the result. For example:

$$\textit{pack}(\texttt{10101}, \texttt{01101}) \Longrightarrow \texttt{011}$$

A function *unpack* is defined as *unpack* : $\mathcal{B}^M \times \mathcal{B}^L \rightarrow \mathcal{B}^L$, where $M \leq L$. *unpack*(x, m) takes all the bits in x and unpacks them, one bit in each position that corresponds to a `1` in m, where m : *bc*$(m) = M$. All unset bits remain `0`. For example:

$$\textit{unpack}(\texttt{011}, \texttt{01101}) \Longrightarrow \texttt{00101}$$

Notice that *pack* and *unpack* are not strict inverses of one another since *pack* destroys information about its first argument. Specifically,

$$\textit{pack}(\textit{unpack}(k, m)) = k \qquad \text{and} \qquad \textit{unpack}(\textit{pack}(k, m)) = k \wedge m$$

2.2 Hyperplane Numbering

A hyperplane or schema is the set of strings in $\mathcal{B}^L$. A hyperplane in $\mathcal{B}^L$ can be represented by an L character string composed of characters from the set: `0`'s, `1`'s, and `*`'s. The `0`'s and `1`'s represent the values of bits that occur in fixed bit positions in the members of the hyperplane. The `*`'s represent where either a `0` or `1` may occur in the variable bit positions. In short, in order for a string to be an element of the hyperplane, the bit values in the string must match the location and values found in the fixed bit positions. An example hyperplane h for strings in $\mathcal{B}^7$ might be `**1101*`. An example of a string in h is `1011010`, while `1001010` is not.

A hyperplane with C fixed bit positions defines a set of $2^{(L-C)}$ strings where all possible replacements of the `*`'s have been used. For a hyperplane h, the number of strings in h is denoted $|h|$. In the case of hyperplane h defined to be `**1101*`, $|h| = 8$. If x is a string in the set of strings defined by the hyperplane h, then we say $x \in h$.

The domain of a function $f : \mathcal{B}^L \rightarrow \mathbb{R}$ can be partitioned into nonintersecting hyperplanes. A partition is classically specified by a partition string. This is a string with a `b` in the positions called fixed bit positions and `*`'s in the remaining positions. For example, `*bb**` represents a partition that contains the hyperplanes `*00**`, `*01**`, `*10**`, and `*11**`. A partition with M `b`'s defines a set of 2^M nonintersecting hyperplanes, each composed of 2^{L-M} strings whose union is all of the strings in the domain.

A unique numbering for hyperplanes can be established. Define the *partition mask* for a partition string as a binary string by replacing the `b`'s by `1`'s and `*`'s by `0`'s. Let the partition mask be the string m. We use $h_{m,n}$, $m \in \mathcal{B}^L$, *bc*$(m) = M$, $M \leq L$, $n \in \mathcal{B}^M$ to denote an M order hyperplane in $\mathcal{B}^L$ such that $x \in h_{m,n}$ if and only if

	$h_{m,0}$	$h_{m,1}$	$h_{m,2}$	$h_{m,3}$	$h_{m,4}$	$h_{m,5}$	$h_{m,6}$	$h_{m,7}$
$h_{\overline{m},0}$	00000	00001	00010	00011	01000	01001	01010	01011
$h_{\overline{m},1}$	00100	00101	00110	00111	01100	01101	01110	01111
$h_{\overline{m},2}$	10000	10001	10010	10011	11000	11001	11010	11011
$h_{\overline{m},3}$	10100	10101	10110	10111	11100	11101	11110	11111

Figure 1: A Partition Matrix for $m = \texttt{01011}$.

pack$(x, m) = n$. In this case, the partition mask m selects how the domain is partitioned and the hyperplane number n selects one of the hyperplanes from the partition. This numbering uniquely encodes each hyperplane as a pair of numbers m, n. Each $x \in h_{m,n}$ can be expressed as *unpack*$(n, m) \vee$ *unpack*$(k, \overline{m})$, where k numbers the strings in the hyperplane. Note that $|h_{m,n}| = 2^{bc(\overline{m})}$ and is independent of n.

For example: if $h_{m,n}$ specifies a hyperplane in $\mathcal{B}^7$ and if $n = \texttt{10110}$ and $m = \texttt{1101110}$, then $h_{m,n}$ is a 5^{th} order hyperplane contained in the partition `bb*bbb*`. m specifies the partition `bb*bbb*`, and n specifies the values of the fixed bit positions, `10110`. Therefore $h_{m,n}$ is `10*110*`.

Two important functions α and β can be defined on a hyperplane (Goldberg, 1989b):

$$\alpha(h)[i] = \begin{cases} \texttt{0} & \text{if } h[i] = \texttt{*} \\ \texttt{1} & \text{if } h[i] = \texttt{0 or 1} \end{cases} \qquad \beta(h)[i] = \begin{cases} \texttt{0} & \text{if } h[i] = \texttt{* or 0} \\ \texttt{1} & \text{if } h[i] = \texttt{1} \end{cases}$$

α returns a mask that identifies the fixed bit positions in the hyperplane. β returns a mask that identifies the bit positions that are set to `1`. For the hyperplane $h = \texttt{**1101*}$: $\alpha(h) = \texttt{0011110}$ and $\beta(h) = \texttt{0011010}$.

The partitioning of the domain space by partition mask m is denoted $h_{m,*}$. The partitioning can be viewed as a concatenation of the $2^{bc(m)}$ hyperplanes $h_{m,n}$ represented as row vector of column vectors:

$$\left[\begin{array}{cccc} h_{m,0} & h_{m,1} & h_{m,2} & \ldots \quad h_{m,2^{bc(m)}-1} \end{array} \right]$$

This forms a $2^{bc(\overline{m})} \times 2^{bc(m)}$ partition matrix, H such that if $H_{i,n} = x$, then $i =$ *pack*$(x, \overline{m})$ and $n =$ *pack*(x, m), where $\overline{m}$ is the complement of m. For example, a function over $\mathcal{B}^5$ with a partition of `b*b**` would give $m = \texttt{10100}, \overline{m} = \texttt{01011}$ and the partition matrix in Figure 1.

In this matrix the bold bits $\{\mathbf{0}, \mathbf{1}\}$ are those determined by *pack*(x, m). All other bits have values determined by *pack*$(x, \overline{m})$. The columns are formed by the hyperplanes $h_{m,*}$, while the rows are formed by the hyperplanes $h_{\overline{m},*}$. The row/column position of string x in the partition matrix is (*pack*$(x, \overline{m})$, *pack*(x, m)).

3 Two Classic Problems

In this section two very important functions in evolutionary computation are discussed. These functions motivate the generalization we will introduce in the section to follow.

NK-landscapes (Kauffman, 1993, 1995) are a popular experimental model for correlated landscapes (Weinberger, 1990) and for modeling gene and species interaction

(Solé and Goodman, 2000). An NK-landscape is a function over N bits $f : \mathcal{B}^N \rightarrow \mathbb{R}$ where K is the number of bits in the string that epistatically interact with each bit. An NK-landscape can be expressed as an average of N functions as follows:

$$f(x) = \frac{1}{N} \sum_{j=0}^{N-1} r_j(\mathit{pack}(x, m_j))$$

Each $r_j : \mathcal{B}^{K+1} \rightarrow [0, 1]$ is an evaluation function that gives a partial fitness based on a subset of bits. In the case of an NK-landscape the r_j are each defined by a table of random numbers in the range $[0, 1]$. The subset of bits is formed by the value of the j^{th} bit and the K bits it interacts with. Therefore, K must fall in the range $[0, N-1]$. Each subset represents the K bits that epistatically interact with the j^{th} bit. To compute the whole function f, N masks, $m_j : \mathit{bc}(m_j) = K + 1$, are used to select N subsets of bits. Each j^{th} bit may interact with a *different* set of K bits. In the expression above, the *pack* function uses the masks to select the subsets of bits and generate the arguments for the interaction functions r_j.

One of the nice features of NK-landscapes is that K acts as a tunable ruggedness control. When $K = 0$, a bitwise linear function is generated. The resulting landscape is the average of the weights associated with each bit and hence, assuming a Hamming neighborhood, the function is highly correlated and relatively smooth. When $K = N - 1$, the function appears random and uncorrelated. Note that NK-landscapes are technically not landscapes but rather functions, since no neighborhood description has been specified for the domain elements (Jones, 1995a, 1995b). However in most cases the literature assumes a one bit Hamming distance neighborhood in the domain to create a landscape.

The second classic type of problem is based on the k-satisfiability or kSAT problem. kSAT problems are a commonly used testbed for search algorithms (Hogg et al., 1996; De Jong et al., 1997). These problems are logical expressions in conjunctive normal form (CNF) with k variables (some of which may be negated) in each disjunctive clause. An expression is satisfiable if there exists an assignment of truth values to the variables that will make the logical expression true. The problem is to decide if an expression is satisfiable or not. When stochastic search algorithms are applied to kSAT problems they need a gradient to climb that is not provided by a range for the function that can only be true or false. So rather than using logical ANDing to combine disjunctive clauses, the Boolean evaluations of individual clauses are summed with *true* being 1 and *false* being 0. The maximum equal to the number of clauses is sought to solve the problem. This function is called MAXSAT (Papadimitriou, 1994).

If C is the number of disjunctive clauses in the CNF, the MAXSAT function of N variables $f : \mathcal{B}^N \rightarrow \mathbb{R}$ is:

$$f(x) = \sum_{j=0}^{C-1} c_j(\mathit{pack}(x, m_j))$$

where $x \in \mathcal{B}^N$ represents the true/false assignment to each of the N Boolean variables in the problem. Each c_j represents a Boolean evaluation function applied to the disjunctive clause that uses the k variables selected by mask $m_j : \mathit{bc}(m_j) = k$ using the *pack* function. In this case $c_j : \mathcal{B}^k \rightarrow \mathcal{B}$.

4 Embedded Landscapes as a Generalization

Embedded landscapes are a useful generalization of these two classes of functions and can be used to decompose any function. An embedded landscape $f : \mathcal{B}^N \rightarrow \mathbb{R}$ can be expressed as the sum of P embedded functions:

$$f(x) = \sum_{j=1}^{P} g_j(\mathit{pack}(x, m_j))$$

There are no restrictions on the number of functions P or the number of `1` bits in each interaction mask $m_j \in \mathcal{B}^N$ or the values returned by the interaction functions $g_j : \mathcal{B}^{bc(m_j)} \rightarrow \mathbb{R}$. The term "embedded landscape" comes from the idea that the g_j are often of lower dimension than N and are embedded in the higher dimensional space via the *pack* function.

Embedded landscapes allow us to independently control several important aspects of a function. By adjusting the value P, we control the number of subfunctions defined over different partitions of the search space. We can, for example, study functions with a variety of densities and distributions of subfunctions. With m_j we can control the exact overlap between subfunction domains. For example, this could be used to study interaction and competitions between different partitions of the search space during genetic search. The subfunctions g_j control the ordering by fitness of hyperplanes in each partition defined by m_j. They also allow us to compare functions constructed out of simple basic classes of smaller subfunctions such as constant, random, linear, or unimodal functions. Although all of this can be done without invoking embedded landscapes, the model provides a ready-made conceptual and mathematical model with several useful orthogonal "knobs" for adjusting function structure.

It is apparent from the definition that both NK-landscapes and MAXSAT problems are forms of embedded landscapes. Embedded landscapes also have the potential to represent many other combinatorial problems that are the sum of the effects of a limited number of interactions between objects such as gene and species interaction, relationships in system models, and functions that are the sum of terms that involve subsets of the chromosome.

Let's look at an example of how a MAXSAT problem and an NK-landscape can both be expressed as an embedded landscape. Let $v = [A, B, C, D]$ be the vector of variables and vector $x \in \mathcal{B}^4$ be an assignment of Boolean values to the variables. Consider the following the MAX2SAT problem:

$$(A \vee \overline{B}) + (B \vee \overline{C}) + (\overline{A} \vee C) + (\overline{B} \vee \overline{D})$$

and an NK-landscape with $N = 4$ and $K = 1$ with the following bit interactions:

$$m_0 = \texttt{1100}, m_1 = \texttt{0110}, m_2 = \texttt{1010}, m_3 = \texttt{0101}$$

and associated epistatic functions r_0, r_1, r_2, r_3 to be described below.

Both these functions can be represented as embedded landscapes. For convenience, the functions are chosen so they have the same interaction masks. The interaction functions for both problems are defined in the same table below.

As an embedded landscape, the MAX2SAT has each disjunctive clause represented as a interaction function c_i and the value of each logical variable as a bit position in the argument string. Then the MAX2SAT problem SAT : $\mathcal{B}^4 \rightarrow \{0, 1, 2, 3, 4\}$ is

$$\text{SAT}(x) = c_0(\textit{pack}(x, m_0)) + c_1(\textit{pack}(x, m_1)) + c_2(\textit{pack}(x, m_2)) + c_3(\textit{pack}(x, m_3))$$

The NK-landscape NK : $\mathcal{B}^4 \rightarrow [0, 1]$ is

$$\text{NK}(x) = \frac{1}{4}r_0(\textit{pack}(x, m_0)) + \frac{1}{4}r_1(\textit{pack}(x, m_1)) + \frac{1}{4}r_2(\textit{pack}(x, m_2)) + \frac{1}{4}r_3(\textit{pack}(x, m_3)))$$

with the interaction functions as shown in the following table, where y represents the value extracted by the pack function $\textit{pack}(x, m_i)$.

	NK subfunctions				2SAT subfunctions			
y	r_0	r_1	r_2	r_3	c_0	c_1	c_2	c_3
`00`	0.31	0.41	0.59	0.26	1	1	1	1
`01`	0.53	0.58	0.97	0.93	0	0	1	1
`10`	0.23	0.84	0.62	0.64	1	1	0	1
`11`	0.33	0.83	0.27	0.95	1	1	1	0

Altenberg (1994, 1996) first introduced the idea of a generalized NK-landscape with a vector of interaction masks in the form of a matrix. His work mainly deals with random interaction functions. We independently created the same natural generalization in Heckendorn and Whitley (1997) in which we referred to the concept as NKP landscapes. With NKP landscapes we emphasized the generality of the embedded functions. In Heckendorn et al. (1999b) we wanted to further emphasize the embedding concept, so we changed the name from NKP to *embedded landscapes*. We reluctantly chose to use the word "landscape" to show an association with NK-landscapes even though the connectivity of the domain is not specified.

5 An Analysis of Embedded Landscapes

An important characteristic of an embedded landscape is its extremely limited epistatic complexity. In this section we will use Walsh analysis (Bethke, 1981; Heckendorn and Whitley, 1999) to study these two features.

An embedded landscape is a sum of simpler lower dimensional subfunctions. This method of combining subfunctions simplifies the analysis of embedded landscapes. To understand the structure of embedded landscapes we must first understand how each lower dimensional subfunction is embedded in the higher dimension of the landscape.

Walsh coefficients are a direct measure of both the linear and nonlinear epistatic interactions of the bits in a function (Goldberg, 1989a; Reeves and Wright, 1995). Any function $f : \mathcal{B}^L \rightarrow \mathbb{R}$ can be uniquely decomposed into a linear weighted sum of Walsh functions ψ_i, where the weights are the real-valued Walsh coefficients w_i.

$$f(x) = \sum_{i=0}^{2^L-1} w_i \psi_i(x)$$

The mapping from 2^L function values of f to 2^L Walsh coefficients is referred to as the Walsh transform of the function f. The Walsh transform creates an invertible linear transformation between a function expressed as a vector and a vector of Walsh coefficients. In fact:

$$w_j = \frac{1}{2^L} \sum_{x=0}^{2^L-1} f(x)\psi_j(x) \tag{1}$$

The value of the Walsh function ψ_i is dependent only on the bit positions indicated by the 1 bits in i. It is a parity based function defined as follows:

$$\psi_i(x) = (-1)^{bc(i \wedge x)}$$

where the exponent is the bit count of the bitwise AND of the subscript and argument to the function. The result is that all Walsh functions return either $+1$ or -1. Note that:

- $\psi_0(x) = \psi_x(0) = 1 \quad \forall x$
- $\psi_i(j) = \psi_j(i)$
- $\psi_i(pack(j, m)) = \psi_{unpack(i,m)}(j)$

If the i^{th} Walsh coefficient w_i is zero, then we can conclude that *exactly* the bit combination identified by the 1 bits in the index i makes **no** contribution to the value of the function. For example: if $w_7 = 0$ then the three bits indicated by the index expressed in binary do not epistatically interact. Note that this does not prevent subsets and super sets of bits from interacting; that is, it may be the case that w_3, w_5, w_6, and w_{15} are nonzero. The order of the Walsh coefficient is the number of 1 bits in the index. A measure that uses Walsh coefficients to quantify the maximum level of epistasis in a function is Ω, which measures the largest number of bits among which there are epistatic interactions:

$$\Omega(f) = \max_{w_i \neq 0} bc(i)$$

5.1 Function Embedding

In this section we show how embedding a single function in a higher dimensional space imposes a relation between the Walsh coefficients of the two functions. Let $g(x) : \mathcal{B}^M \rightarrow \mathbb{R}$ and $f(x) : \mathcal{B}^L \rightarrow \mathbb{R}$ with $M \leq L$. g is said to be an *embedding* of g in f if there is a mask $m : bc(m) = M$ and $m \in \mathcal{B}^L$ such that $f(x) = g(pack(x, m)) \quad \forall x \in \mathcal{B}^L$. This is an embedding of a lower dimension function g in a higher dimension function f by setting the fitness of every string in the hyperplane $h_{m,x}$ of f to $g(x)$. That is, the value of the function f is only dependent on the bits in the positions indicated by the interaction mask m. See Figure 2.

If g is in $\mathcal{B}^M$ and f is in $\mathcal{B}^L$, then the Walsh polynomial for g has Walsh functions that take M bit arguments, while the Walsh polynomial for f has Walsh functions that take L bit arguments. Despite this, the following theorem show that the Walsh coefficients for the two functions are the same, and hence the interaction pattern between bits, in a strong sense, is preserved.

THEOREM 1 (Embedding Theorem): *Let $g(x) : \mathcal{B}^M \rightarrow \mathbb{R}$ and $f(x) : \mathcal{B}^L \rightarrow \mathbb{R}$ and g is embedded in f, then*

$$w_i^f = \begin{cases} w^g_{pack(i,m)} & \text{if } i \subseteq m \\ 0 & \text{otherwise} \end{cases}$$

where w^g and w^f are Walsh coefficients for the g and f functions, respectively, and $i, m \in \mathcal{B}^L; bc(m) = M$.

	$h_{m,0}$	$h_{m,1}$	$h_{m,2}$	$\ldots$	$h_{m,\vec{1}}$
$h_{\overline{m},0}$	$g(0)$	$g(1)$	$g(2)$	$\ldots$	$g(2^{bc(m)}-1)$
$h_{\overline{m},1}$	$g(0)$	$g(1)$	$g(2)$	$\ldots$	$g(2^{bc(m)}-1)$
$h_{\overline{m},2}$	$g(0)$	$g(1)$	$g(2)$	$\ldots$	$g(2^{bc(m)}-1)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$
$h_{\overline{m},\vec{1}}$	$g(0)$	$g(1)$	$g(2)$	$\ldots$	$g(2^{bc(m)}-1)$

Figure 2: The function values for a single embedded landscape using an interaction mask of m.

PROOF:

$$\begin{aligned}
w_i^f &= \tfrac{1}{2^L}\textstyle\sum_{x=0}^{2^L-1} f(x)\psi_x(i)\\
&= \tfrac{1}{2^L}\textstyle\sum_{x=0}^{2^L-1} g(\mathit{pack}(x,m))\psi_x(i)\\
&\quad \text{we change the dimension of the function here:}\\
&= \tfrac{1}{2^L}\textstyle\sum_{j=0}^{2^M-1}\sum_{x:\mathit{pack}(x,m)=j} g(\mathit{pack}(x,m))\psi_x(i)\\
&= \tfrac{1}{2^L}\textstyle\sum_{j=0}^{2^M-1}\sum_{x:\mathit{pack}(x,m)=j} g(j)\psi_x(i)\\
&= \tfrac{1}{2^L}\textstyle\sum_{j=0}^{2^M-1} g(j)\sum_{x:\mathit{pack}(x,m)=j}\psi_x(i)\\
&= \tfrac{1}{2^L}\textstyle\sum_{j=0}^{2^M-1} g(j)\sum_{x\in h_{m,j}}\psi_x(i)\\
&= \tfrac{1}{2^L}\textstyle\sum_{j=0}^{2^M-1} g(j)\sum_{x\in h_{m,j}}\psi_i(x)
\end{aligned}$$

We know from the Balanced Sum Theorem for Hyperplanes (Goldberg, 1989a) that

$$\sum_{x\in h_{m,j}}\psi_i(x) = \begin{cases} 0 & \text{if } i \not\subseteq m \\ \psi_i(\beta(h_{m,j}))|h_{m,j}| & \text{if } i \subseteq m \end{cases}$$

where $|h_{m,j}|$ is the number of strings in $h_{m,j}$.

Case 1: Assume $i \not\subseteq m$:

$$\begin{aligned}
w_i^f &= \tfrac{1}{2^L}\textstyle\sum_{j=0}^{2^M-1} g(j)\sum_{x\in h_{m,j}}\psi_i(x)\\
&= \tfrac{1}{2^L}\textstyle\sum_{j=0}^{2^M-1} g(j)0\\
&= 0
\end{aligned}$$

Case 2: Assume $i \subseteq m$:

$$\begin{aligned}
w_i^f &= \tfrac{1}{2^L}\textstyle\sum_{j=0}^{2^M-1} g(j)\sum_{x\in h_{m,j}}\psi_i(x)\\
&= \tfrac{1}{2^L}\textstyle\sum_{j=0}^{2^M-1} g(j)\psi_i(\beta(h_{m,j}))|h_{m,j}|\\
&= \tfrac{1}{2^L}\textstyle\sum_{j=0}^{2^M-1} g(j)\psi_i(\beta(h_{m,j}))2^{L-M}\\
&= \tfrac{1}{2^M}\textstyle\sum_{j=0}^{2^M-1} g(j)\psi_i(\beta(h_{m,j}))\\
&= \tfrac{1}{2^M}\textstyle\sum_{j=0}^{2^M-1} g(j)\psi_i(\mathit{unpack}(j,m))\\
&= \tfrac{1}{2^M}\textstyle\sum_{j=0}^{2^M-1} g(j)\psi_{\mathit{pack}(i,m)}(j)\\
&= w^g_{\mathit{pack}(i,m)}
\end{aligned}$$

	$h_{m,0}$	$h_{m,1}$	$\ldots$	$h_{m,\vec{1}}$
$h_{\overline{m},0}$	$w_{\mathit{unpack}(0,m)}$	$w_{\mathit{unpack}(1,m)}$	$\ldots$	$w_{\mathit{unpack}(\vec{1},m)}$
$h_{\overline{m},1}$	0	0	$\ldots$	0
$h_{\overline{m},2}$	0	0	$\ldots$	0
$\vdots$	$\vdots$	$\vdots$		$\vdots$
$h_{\overline{m},\vec{1}}$	0	0	$\ldots$	0

Figure 3: The location of the nonzero Walsh coefficients for an embedded function with an interaction mask of m.

It is clear that since $i \subseteq m$, the mapping from Walsh coefficients in g to Walsh coefficients in f is 1-1. □

The critical observation is that even though all of the function values in f may be nonzero, only those Walsh coefficients w_i with $i \subseteq m$ can be nonzero. That is, for the Walsh transform of function f, $h_{\overline{m},0}$ contains the only nonzero Walsh coefficients. This leads to a second important observation: embedding a lower dimensional function, such as g above, in a higher dimensional space, as with function f above, neither increases the number of nonzero Walsh coefficients nor the maximum level of epistasis.

In terms of the matrix of Walsh coefficients, applying the Walsh transform to the earlier matrix (Figure 2) for the embedded function g yields the distribution of Walsh coefficients we see in Figure 3, where the w_i are the Walsh coefficients for g. (It is critical to note that the indices of w shown here are taken from f; their indices in g are simply the first argument to *unpack*.)

Because an embedded landscape is a sum of subfunctions and the Walsh transform is a linear transformation, we know the i^{th} Walsh coefficient of the embedded landscape f with subfunctions g_j can be computed by:

$$w_i^f = \sum_{j=1}^{P} w_{\mathit{pack}(i,m_j)}^{g_j} \text{ if } i \subseteq m_j$$

That is, given i, we need only sum up the Walsh coefficients for the subfunctions g_j when $i \subseteq m_j$. And in general, for a fixed value of k, the Walsh coefficients of an embedded landscape can be computed in polynomial time relative to 2^k, provided the subfunction masks are known in advance. Since k is fixed independently of L, the Walsh coefficients can be computed in polynomial time relative to the function size in number of bits. It is also clear that only a polynomial number of Walsh coefficients are nonzero.

Epistatically bounded functions are functions in which the number of epistatically interacting bits is bound above by k. Technically, this class of functions can be modeled by embedded landscapes in which $\mathit{bc}(m_j) \leq k, \forall j$. However, the number of subfunctions necessary to model an arbitrary L bit function whose epistatic interactions are bound by k bits is $\binom{L}{k}$ where most of the values of the subfunction provide redundant epistatic information. This makes embedded landscapes impractical for the most general version of this class of function.

Kargupta and Park (1999) have shown that for epistatically bounded functions f, where $\Omega(f) = k$ is known, all the Walsh coefficients can be computed in polynomial time relative to 2^k. Again, this can be considered polynomial time in cases where the epistatic bound k is a fixed constant independent of L. As in the case of embedded landscapes, there are a polynomial number of nonzero Walsh coefficients.

Kargupta's algorithm uses hyperplane averages (that is, the average value of the function for all values in the hyperplane (see the notation section)) to probe a function's epistatic structure. Using this technique the algorithm does not need to know *a priori* the epistatic structure of the function, only the limit of the epistasis Ω.

6 Epistatic Structure

For an unconstrained function of N bits the number of Walsh coefficients that can be nonzero is all of them or 2^N. As constraints are applied to the function, patterns appear in the Walsh coefficients. One way to look for patterns is to look at which Walsh coefficients can be nonzero and which must be zero. In this section we will look at a variety of ways of measuring the coverage of the available Walsh coefficients with nonzero values.

Knowing the maximum epistasis is not as informative as knowing the distribution of nonzero Walsh functions called the Walsh distribution. If we group the Walsh coefficients w_i by the order of the coefficient, that is $bc(i)$, then the number of Walsh coefficients of each order forms a binomial distribution as seen by the solid line in Figure 4. This is measured as a vector $\vec{\mathcal{K}}$ where $\mathcal{K}_i$, the i^{th} Walsh count, is a count of the number of nonzero Walsh coefficients of order i.

The allowable nonzero Walsh coefficients for embedded functions can be far less than for an arbitrary function. Consider a single embedded function f with interaction mask m. Since the only nonzero Walsh coefficients in f have an index whose 1 bits are contained entirely in the mask m, there are at most $2^{bc(m)}$ nonzero Walsh coefficients for f. All the other Walsh coefficients must be zero. This is reasonable since bits outside of mask m should have no effect on the value of the higher dimensional function. Therefore for values of $bc(m) \ll L$ the ratio of nonzero Walsh coefficients to the total number of available Walsh coefficients to describe the function becomes exponentially small.

If we look within each order of coefficient, then if g is embedded in f using mask m, there are at most $\binom{bc(m)}{k}$ nonzero Walsh coefficients of order k. Each subfunction generates its own binomial distribution of potentially nonzero Walsh coefficients in which there will be one Walsh coefficient of order $bc(m)$, namely w_m, and $bc(m)$ of order $bc(m) - 1$ and so on to $bc(m)$ of order 1 and 1 of order 0. See Figure 4.

The distribution of nonzero Walsh coefficients of f is the joining of the sets of overlapping distributions of the subfunctions. But even adding a polynomial number of subfunctions will not allow the number of nonzero Walsh coefficients to become comparable to exponential number of available Walsh coefficients. This strongly constrains the complexity of the function.

THEOREM 2 (Walsh Distribution Limit for Embedded Landscapes): *If f is an N bit embedded landscape with P embedded subfunctions g_i, then*

$$\mathcal{K}_R \;\leq\; \min\left(\binom{N}{R}, \sum_{i=1}^{P}\binom{\Omega(g_i)}{R}\right) \;\leq\; \min\left(\binom{N}{R}, P\binom{\Omega(f)}{R}\right) \tag{2}$$

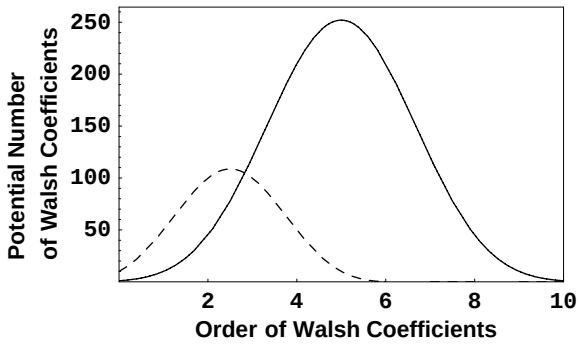

Figure 4: Walsh coefficient distributions for an arbitrary 10 bit function (solid line) and for a 10 bit fixed embedded landscape (dashed line) with 10 subfunctions each with $\Omega = 5$.

PROOF: The first upper bound is true because the number of nonzero Walsh coefficients can be overestimated by summing the number of Walsh coefficients for the subfunctions as if they were completely independent. The resulting value must be clipped by a $\min$ function to the total number of possible nonzero Walsh coefficients for a N bit function.

The second upper bound is true because each subfunction could be more coarsely bounded by the most epistatic subfunction in the landscape, which is the same as epistasis of the whole function. □

In practice there is a lot of duplication when embedding masks overlap since any number of interaction masks may imply the same Walsh coefficient is nonzero. The result is that the number of nonzero Walsh coefficients of order R is considerably less than our proposed upper bounds.

Let's take an example. In most circumstances where embedded landscapes are used as test functions or in cases where component parts of a combinatorial problem have the same degree of interaction, the size of the subfunctions is constant, that is

$$\Omega(g_i) = \Omega(f) \qquad \forall i \in \{1, 2, \ldots, P\}$$

I will call these *fixed size embedded landscapes*. For fixed size embedded landscapes the second upper bound in the Walsh Limit Theorem above applies with equal accuracy as the first upper bound. Fixed size embedded landscapes are just as general as nonfixed size embedded landscapes if the subfunctions themselves are allowed to be functions of limited epistasis. Fixing the size at a value K gives us a single figure to talk about size of the function rather than considering a vector of the dimensions of all of the subfunctions. For simplicity, in much of the remainder of this paper we will consider embedded landscapes to be of fixed size.

Consider a fixed size embedded landscape f with $\Omega(f) = K$. The maximum possible number of nonzero Walsh coefficients of order R for an embedded function with P subfunctions is $P\binom{K}{R}$. The ratio of this limit to the total number of possible order R

Table 1: The upper bound of number of nonzero Walsh coefficients in a general 10 bit function vs. the number in a 10 bit fixed embedded landscape with 10 subfunctions broken down By order R and $\Omega = K$. When a ratio is given, the numerator is the upper bound on the number of nonzero Walsh coefficients for the embedded landscape and the denominator is the maximum possible for any function of order N.

$K \rightarrow$ $\downarrow R$	1	2	3	4	5	6	7	8	9	10
0	1	1	1	1	1	1	1	1	1	1
1	10	10	10	10	10	10	10	10	10	10
2	-	10/45	30/45	45	45	45	45	45	45	45
3	-	-	10/120	40/120	100/120	120	120	120	120	120
4	-	-	-	10/210	50/210	150/210	210	210	210	210
5	-	-	-	-	10/252	60/252	210/252	252	252	252
6	-	-	-	-	-	10/210	70/210	210	210	210
7	-	-	-	-	-	-	10/120	80/120	120	120
8	-	-	-	-	-	-	-	10/45	45	45
9	-	-	-	-	-	-	-	-	10	10
10	-	-	-	-	-	-	-	-	-	1

Walsh coefficients $\binom{N}{R}$ is a good indicator of sparseness of the coverage by the embedded landscape.

Table 1 shows a comparison of the maximum number of nonzero Walsh coefficients for embedded landscapes of $\Omega(f) = K$ with $N = 10, P = 10$ and landscapes with $N = 10$, which have all Walsh coefficients of order K or less nonzero. The columns represent the varying values of K. The rows represent the various orders R of Walsh coefficients for the function. Therefore each column represents the Walsh distribution for a different level of maximum epistasis in an embedded landscape versus an epistatically limited landscape. Wherever a single number occurs in the table, the embedded landscape may have the maximum number of nonzero Walsh coefficients for that order. When a ratio is given, the numerator is the upper bound on the number of nonzero Walsh coefficients for the embedded landscape and the denominator is the maximum possible for any function of order N. For instance, for an embedded landscape with $N = 10$ and $K = 5$, there are $\binom{10}{5} = 252$ Walsh coefficients of order $R = 5$ but at most $10\binom{6}{5} = 60$ of them can be nonzero in the embedded landscape for $P = 10$.

Another way to look at this is presented in Figure 4. Here the X-axis represents the order R of the Walsh coefficients. The Y-axis represents the number of possible Walsh coefficients of that order. The solid line represents the maximum number of possible nonzero Walsh coefficients for an arbitrary function over the 10 bit domain. The standard continuous Gaussian approximation for a binomial distribution is used to ease visualization. The dashed line shows the maximum number of nonzero Walsh coefficients for a 10 bit embedded landscape with 10 subfunctions and the Ω for each subfunction of 5. This graph shows that the greatest number of possible Walsh coefficients are of order $N/2$ or 5. When the 10 functions are summed, the coverage is increased 10 times and the left tail of the upper dashed curve arches above the solid curve. It is the minimum of these two curves, as per Equation 2, that limits the total number of Walsh coefficients for the embedded landscape. For larger Ω the curve increases in size, and the mean moves to the right until all possible functions of order N are covered.

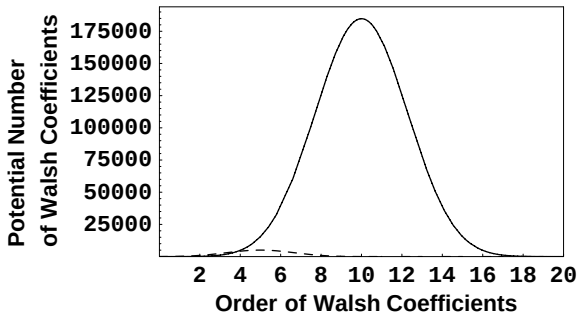

Figure 5: Walsh coefficient distributions for an arbitrary 20 bit function (solid line) and for a 20 bit fixed embedded landscape (dashed line) with 20 subfunctions each with $\Omega = 10$.

As N increases, the discrepancy between embedded landscapes and arbitrary functions of $\Omega = K$ becomes increasingly pronounced. In Figure 5 the solid curve is the potential number of nonzero Walsh coefficients for a 20 bit function while the small dashed curve at the bottom of the graph is the distribution for an embedded landscape of 20 functions with $\Omega = 10$.

Figure 6(a) shows the upper bound for the number of nonzero Walsh coefficients for an embedded landscape with $N = 100$ and various values of K. The lines marked with cross, diamond, square, and triangle indicate values of P equal to 10^2, 10^3, 10^4, and 10^5, respectively. It is clear that for even large P, relative to N, the function is in large part devoid of epistasis. Only when $K \rightarrow N$ does the amount of available epistatic interactions approach the number available. The transition is exponentially abrupt. The *coverage* is the ratio of the number of possible nonzero Walsh coefficients to available Walsh coefficients. Figure 6(b) displays the coverage for the same functions. This figure emphasizes the abruptness of the transition from sparseness to fully epistatic in the linearity of the coverage as $K \rightarrow N$ when measured on a $\log_{10}$ scale. The sparseness for $N = 100$ is also emphasized in this figure since for $P = 100$ and $K = 34$ the ratio is about $1.514E - 15$.

The intersection of the maximum possible coverage of P independent K order functions and the limit imposed by the general N bit function. That is, the intersection of the upper dashed line and the solid line on the graph as measured in bits of interaction as in Figure 4. Note that the intersection remains small for most values of K and only nears N when K nears N. This can be seen in Figure 7. This allows the maximum possible number of nonzero Walsh coefficients for an embedded landscape to be approximated by $N2^K$ for most values of K except near 1 and N. For $K \geq N/2$ the number of available Walsh coefficients of a given order or less is approximately 2^{N-1}, therefore for $K \geq N/2$ and not near N an upper bound for the coverage is $N2^K/2^{N-1}$ or $2N/2^{N-K}$. As K decreases from $N/2$ the number of coefficients for an embedded landscape continues to decrease by powers of two while the number of available coefficients with K or fewer bits initially decreases more slowly. The result is that for most $K < N/2$ where K is not near 0 the value of the coverage is less than it is at $K = N/2$ giving an upper bound of $2N/2^{N/2}$.

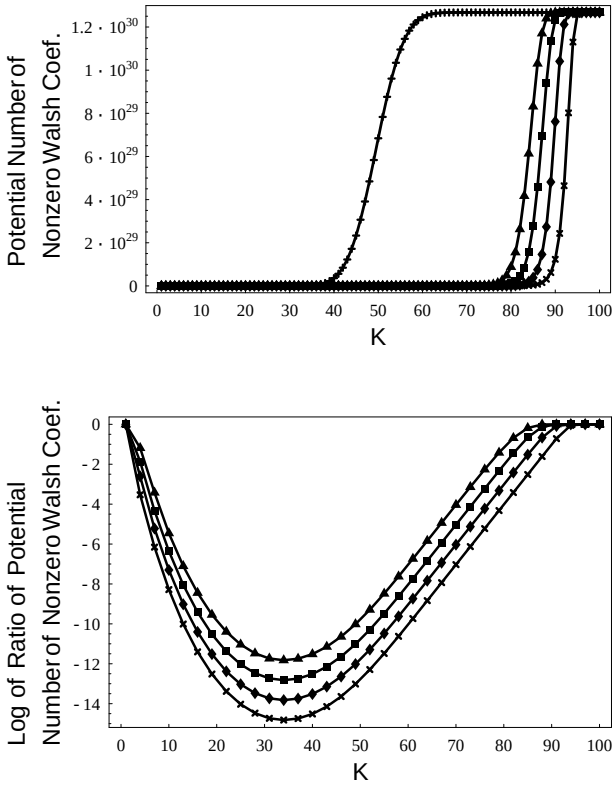

Figure 6: a) The upper bound of the number of nonzero Walsh coefficients for embedded landscapes with $N = 100$ and $P = 100$(cross), 10^3(diamond), 10^4(square), 10^5(triangle) and the maximum number of potential nonzero Walsh coefficients for a function whose maximum epistasis is K (plus). b) The $\log_{10}$ of the ratio of the number of nonzero Walsh coefficients in the same embedded landscapes vs. the total number of available Walsh coefficients.

These upper bounds for a single function embedding suggest that for most orders of Walsh coefficients (not near 0 or N) in an embedded landscape the number of Walsh coefficients that are nonzero is exponentially small. For small K or small P there will not be enough subfunctions or they won't be large enough to compensate for the exponentially diminished number of nonzero Walsh coefficients provided by the subfunctions. This means the functions represent an exceedingly simplified subset of the possible functions as measured by the number of possible ways bits could epistatically interact in the function.

6.1 Computing the Walsh Distribution by Masks

In the previous section we showed an upper bound for the distribution of nonzero Walsh coefficients. This assumed that none of the N bit positions has a `1` in more than one of the interaction masks m_b. If we know the set of masks, we can exactly compute the distribution of Walsh coefficients. This gives us some insight on how overlapping regions of bit interaction interfere to reduce the diversity of functions as seen by the

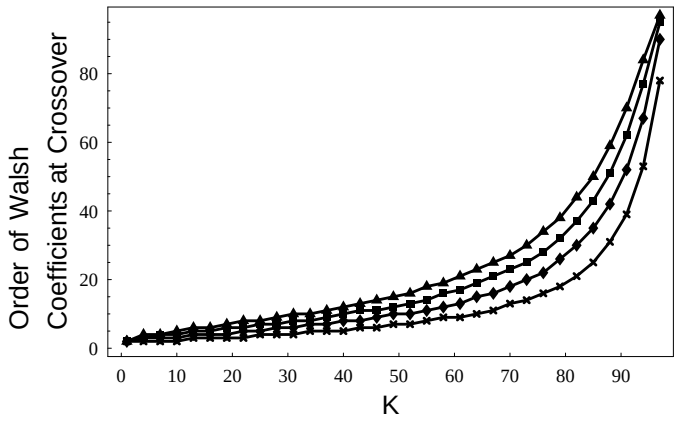

Figure 7: The intersection point for $N = 100$ and $P = 100$(cross), 10^3(diamond), 10^4(square), 10^5(triangle).

Walsh coefficients.

We know from the Embedding Theorem that the nonzero Walsh coefficients for the embedding of a function must have indices that are contained in the mask used for the embedding. Each interaction mask can therefore generate 2^{K+1} nonzero Walsh coefficients in the final function, but only if there is not any overlapping between masks.

But suppose there is overlapping. Let's examine the case of a function f that is composed of the sum of the embeddings of two functions f_1 and f_2 based on corresponding overlapping masks m_1 and m_2 with $bc(m_1) = \delta_1$ and $bc(m_2) = \delta_2$. Assume the masks share δ_{12} bits in common, i.e., $bc(m_1 \wedge m_2) = \delta_{12}$. The maximum number of nonzero Walsh coefficients for function f is the sum of the coefficients covered by the two functions minus the coefficients duplicated by the intersection.

That is, if we know the Walsh counts $\mathcal{K}_i$ for embedded functions g_1 and g_2 then:

$$\mathcal{K}_i^g \leq \mathcal{K}_i^1 + \mathcal{K}_i^2 - \mathcal{K}_i^{12}$$

where $\mathcal{K}_i^1$ is the i^{th} Walsh count for the function generated by an embedding based on m_1, and $\mathcal{K}_i^{12}$ is the Walsh count for a function generated by an embedding based on $m_1 \wedge m_2$.

For example: in Table 2 we assume the function f is composed of the embedding of two functions g_1 and g_2 on the domain $\mathcal{B}^4$ expanded with the respective masks $m_1 =$ `1110001` and $m_2 =$ `0011101`. The first column is the order of the Walsh coefficients being counted. Since the masks are 7 bits long, the domain of f is $\mathcal{B}^7$, and therefore there are 8 different Walsh counts for f and the embeddings of g_1 and g_2. The second column is the distribution of the maximum number of nonzero Walsh coefficients for the embedding of g_1 given that the embedding was done using a mask with 4 bits set. g_2 is similarly represented in column three. The fourth column is the maximum number of nonzero walsh coefficients that are duplicated in the interaction of the two masks. The duplicate counts are subtracted and yield the final column, which is the maximum number of nonzero coefficients for the given masks.

Table 2: Example Walsh count computation.

i	$\mathcal{K}_i^1$	$\mathcal{K}_i^2$	$\mathcal{K}_i^{12}$	$\mathcal{K}_i^f$
0	1	1	1	1
1	4	4	2	6
2	6	6	1	11
3	4	4	0	4
4	1	1	0	1
5	0	0	0	0
6	0	0	0	0
7	0	0	0	0

In general, by using the Inclusion-Exclusion Principle (Niven, 1965) from combinatorics we can see for f based on N subfunctions that are expanded using masks $m_b : b \in \{1, 2, \ldots, N\}$ the distribution of Walsh counts can be calculated as follows:

$$\vec{\mathcal{K}}^f = \sum_{1 \le i \le N} \vec{\mathcal{K}}^i - \sum_{1 \le i < j \le N} \vec{\mathcal{K}}^{ij} + \sum_{1 \le i < j < k \le N} \vec{\mathcal{K}}^{ijk} - \cdots - (-1)^N \vec{\mathcal{K}}^{(allmasks)}$$

The first term in the expression represents the sum of the contributions of each mask if the masks were all disjoint. Each successive sum in the expression is an interaction term for increasing numbers of masks.

Why doesn't this provide a formula for the exact number of nonzero Walsh coefficients? The answer is best illustrated by going back to the two mask example. The Walsh counts do not contain all of the information necessary to determine which Walsh coefficients are covered by both g_1 and g_2 when all possible Walsh coefficients of each order are *not* covered. Of course, an exhaustive listing of which coefficients are covered and which are not by each embedding function will determine the exact coverage.

7 The NP-Completeness of Embedded Landscapes

Although embedded landscapes can be quite limited in complexity, that doesn't make them easy. A decision language L is said to be NP-hard if

$$L' <_p L \text{ for every } L' \in \text{NP}$$

where $L' <_p L$ implies that language L' reduces to L in polynomial time. A language L' reduces to L if every instance of a problem in L' can be re-expressed as an instance of a problem in L (Cormen et al., 1990). Let L_e be the language corresponding to the set of all embedded landscapes. It is well known (Papadimitriou, 1994) that SAT and 3SAT are NP-hard and that

$$\text{SAT} <_p \text{3-SAT} <_p \text{MAX3SAT}.$$

Since by definition the language L_e contains every instance of the language MAX3SAT, it trivially follows that

$$\text{SAT} <_p \text{3-SAT} <_p \text{MAX3SAT} <_p L_e$$

and thus that L_e is NP-hard.

To be NP-complete, a language L must be both

1. NP-hard and
2. $L \in NP$.

Is it true that $L_e \in$ NP? In general, the answer is no. For example, assume that the subfunction g_i takes k bits as input and that k is allowed to vary with N. If, for example, $k = N/2$ then the subfunctions g_i are exponentially large with respect to N. Therefore an arbitrary random function g_i may not have a compact representation; thus, it can only be described using exponential time/space with respect to N. In this case, the evaluation function has exponential cost. Thus, even a nondeterministic Turing machine cannot find or verify a solution in polynomial time. In the general case, embedded landscapes are NP-hard, but not NP-complete. In this sense embedded landscapes are more difficult than NP-complete problems.

The same argument holds for NK landscapes. If an NK landscape allows K to vary with N, then NK landscapes are not in the complexity class NP. Yet Weinberger (1996) proves that NK landscapes are NP-complete for $K \geq 3$. How is this possible? What Weinberger actually shows is that for any specific fixed K, NK landscapes are NP-complete. If K is fixed and not allowed to vary with N, then the size of each subfunctions g_i is of size 2^{K+1}. This may be a large or small constant, but it is a constant. The problem description for NK landscapes then has complexity $\mathcal{O}(N)$.

The same restriction can be placed on embedded landscapes. In practice, the only embedded landscapes that can be generated in polynomial time are such that

1. k is bounded by some constant
2. the *number* of subfunctions P is polynomial in N.

Note that when k is fixed, the number of possible subfunctions is automatically polynomial with respect to N. If every subfunction g_i uses exactly k bits then every embedded landscape of fixed k can be constructed using at most $P = \binom{N}{k}$ subfunctions.

Thus, when k is fixed, embedded landscapes have a polynomial time description. We will refer to the language over the subset of embedded landscapes with polynomial time descriptions as L_{pe}. An optimal solution to a problem in L_{pe} can be found or verified in polynomial time by a nondeterministic Turing machine. The verification is the same as the verification for *Traveling Salesperson Problems* (Cormen et al., 1990): we must be told in advance the evaluation of the optimal solution. We then verify the solution we are given has that particular evaluation – which we can do since the evaluation function is polynomial. Therefore $L_{pe} \in$ NP and it follows that:

1. L_e is NP-hard but not NP-complete since MAX3SAT $<_p L_e$ and $L_e \notin$ NP.
2. L_{pe} is NP-hard and NP-complete since MAX3SAT $<_p L_{pe}$ and $L_{pe} \in$ NP.

We can only be certain that an exact polynomial time Walsh analysis exists if those problems are taken from the language L_{pe}. Obviously, languages in L_e that do not allow a polynomial time description do not allow a polynomial time Walsh analysis.

Note that it is possible for all NP-complete problems to be expressed as MAXSAT problems. In particular, as we will show, we can do a polynomial time Walsh analysis on MAX3SAT in time proportional to the problem description. This means that we can also exactly compute all schema averages up to any fixed order denoted by q. By fixing q we again restrict our attention to a polynomial number of schema averages. Yet, if the complexity classes $P \neq$ NP then performing an exact Walsh analysis and exactly having all static schema averages for all schema up to order q is not sufficient to guarantee that one can infer the global optimum.

8 Summary Statistics

Summary statistics (mean, variance, skew, etc.) tell us about the location, spread, symmetry, etc. of a distribution of numbers. When applied to fitness functions they give us statistical information on the expected value of a random sample, as in an initial population, and how much we can expect elements in our sample, on average, to deviate in both directions from the mean. Furthermore, it is widely believed (Whitley et al., 1995; Holland, 1975) that GAs work by a form of implicit parallelism. That is, all hyperplanes in a given population compete in parallel for representation in succeeding populations. The effectiveness of this competition is limited by several factors including the ability of a sample of elements from the hyperplane to represent the true fitness of the entire hyperplane. Summary statistics about a specific hyperplane would be useful in determining how accurately a sampling of elements from a hyperplane may statistical "stand in" for that hyperplane in a population.

Clearly, directly computing the summary statistics for arbitrary fitness functions would require exponential time relative to the size of the domain in bits. The same could be said for low order hyperplanes. In this section we show that three kinds of statistical moments can be computed in polynomial time for any function in which there are at most a polynomially bounded number of nonzero Walsh coefficients such as embedded landscapes.

r^{th} moments are essential to computing summary statistics. Rana-Stevens suggested that the r^{th} moments of embedded landscapes might also be computed in polynomial time. Subsequently, we showed (Heckendorn et al., 1999a) that this was indeed the case. This means that since we can compute all of the Walsh coefficients as well as the summary statistics for embedded landscapes, which are generally NP-complete for $k \geq 2$ in polynomial time we can say that either NP is P, or knowing the Walsh coefficients and summary statistics for a polynomial number of hyperplanes is insufficient to discover the optimum of the problem in polynomial time (Heckendorn et al., 1999a). This is an important result on the limits of the usefulness of epistatic information. Our proofs hinge only on the facts that, for the functions of interest, there are at most a polynomial number of nonzero Walsh coefficients and that their values can be discovered in polynomial time. With the introduction of Kargupta's algorithm our results extend to all epistatically bounded functions. In this section, I will also show how summary statistics such as skew and kurtosis can also be computed from the Walsh coefficients by using a general formula for computing the r^{th} moment for any function where all the nonzero Walsh coefficients are known.

THEOREM 3 (Moment about Function Mean): *The r^{th} moment μ_r for a fitness function $f : \mathcal{B}^L \rightarrow \mathbb{R}$ whose Walsh coefficients are w_j is*

$$\mu_r = \sum_{a_1 \oplus a_2 \oplus \ldots \oplus a_r = 0} w_{a_1} w_{a_2} \ldots w_{a_r}, \qquad a_i \neq 0 \; \forall \; i$$

where $\oplus$ is the EXCLUSIVE-OR *operator.*

PROOF: Given the mean μ, the formula used to compute the r^{th} moment μ_r for a discrete random variable X is:

$$\mu_r \; = \; E[(X - \mu)^r] \; = \; \sum_{x \in X} (x - \mu)^r p(x)$$

where $p(x)$ is the probability of x occurring (Mendenhall, 1967). We can consider $p(x) = \frac{1}{2^L}$ since we are enumerating a function over all L bit binary strings. The function then becomes:

$$\mu_r = \sum_{x \in X} \frac{(x-\mu)^r}{2^L}$$

If X represents a real valued function over an L bit domain, then:

$$\mu_r = \frac{1}{2^L} \sum_{x=0}^{2^L-1} (f(x)-\mu)^r$$

We can substitute for f with the linear Walsh representation of f:

$$\mu_r = \frac{1}{2^L} \sum_{x=0}^{2^L-1} \left(\sum_{i=0}^{2^L-1} w_i \psi_i(x) - \mu \right)^r$$

Since $\psi_0(x) = 1 \; \forall x$, we see from Equation 1 that Walsh coefficient w_0 is the mean of all fitnesses. Therefore,

$$\mu_r = \frac{1}{2^L} \sum_{x=0}^{2^L-1} \left(\sum_{i=1}^{2^L-1} w_i \psi_i(x) \right)^r$$

We can now expand the exponential creating a set of r indices a_j where $a_j \in \mathcal{B}^L$:

$$\mu_r = \frac{1}{2^L} \sum_{x=0}^{2^L-1} \left(\sum_{a_1=1}^{2^L-1} w_{a_1} \psi_{a_1}(x) \right) \left(\sum_{a_2=1}^{2^L-1} w_{a_2} \psi_{a_2}(x) \right) \ldots \left(\sum_{a_r=1}^{2^L-1} w_{a_r} \psi_{a_r}(x) \right)$$

Since the Walsh coefficients do not depend on x, the formula can be rewritten as:

$$\mu_r = \frac{1}{2^L} \sum_{a_1=1}^{2^L-1} \sum_{a_2=1}^{2^L-1} \cdots \sum_{a_r=1}^{2^L-1} w_{a_1} w_{a_2} \ldots w_{a_r} \sum_{x=0}^{2^L-1} \psi_{a_1}(x) \psi_{a_2}(x) \ldots \psi_{a_r}(x)$$

For arbitrary p and q: $\psi_p(x)\psi_q(x) = \psi_{p \oplus q}(x)$, therefore:

$$\mu_r = \frac{1}{2^L} \sum_{a_1=1}^{2^L-1} \sum_{a_2=1}^{2^L-1} \cdots \sum_{a_r=1}^{2^L-1} w_{a_1} w_{a_2} \ldots w_{a_r} \sum_{x=0}^{2^L-1} \psi_{a_1 \oplus a_2 \oplus .. \oplus a_r}(x)$$

Now using the fact that:

$$\sum_{x=0}^{2^L-1} \psi_i(x) = \begin{cases} 0 & \text{if } i \neq 0 \\ 2^L & \text{if } i = 0 \end{cases}$$

we see that only when $a_1 \oplus a_2 \oplus \ldots \oplus a_r = 0$ is the inner sum nonzero. Therefore,

$$\mu_r = \frac{1}{2^L} \sum_{a_1 \oplus a_2 \oplus \ldots \oplus a_r = 0} w_{a_1} w_{a_2} \ldots w_{a_r} 2^L, \qquad a_i \neq 0 \; \forall \; i$$

$$= \sum_{a_1 \oplus a_2 \oplus \ldots \oplus a_r = 0} w_{a_1} w_{a_2} \ldots w_{a_r}, \qquad a_i \neq 0 \; \forall \, i \tag{3}$$

□

To summarize, given the set of nonzero Walsh coefficients, we can compute the r^{th} moment for the fitness distribution using products of the Walsh coefficients such that the EXCLUSIVE-OR of the indices is zero.

This formula allows us to compute the variance, skew, and kurtosis for any fitness distribution provided we are given the Walsh coefficients.

$$\text{variance} = \mu_2 = \sigma^2 \qquad \text{skew} = \frac{\mu_3}{\sigma^3} \qquad \text{kurtosis} = \frac{\mu_4}{\sigma^4}$$

For example, since $a_1 \oplus a_2 = 0$ if and only if $a_1 = a_2$, then the variance for any function can be computed

$$\sum_{i=1}^{2^L - 1} w_i w_i$$

Of course, the computation of the moment around the mean, if done directly, would take $\mathcal{O}(2^{Lr})$ time. However, in the case of functions with only a polynomial number of nonzero Walsh coefficients, the nonzero coefficients are easily enumerated. Only the nonzero coefficients need be considered in the moment calculations.

In the case of embedded landscapes, the Walsh coefficients are computed for each subfunction in $\mathcal{O}(k2^k)$ using a fast Walsh transform (Goldberg, 1989a). So for P functions the Walsh coefficients can be computed in $\mathcal{O}(Pk2^k)$, where P is bounded by $\binom{L}{k}$ for L bit functions (Heckendorn et al., 1999b). Even though the computation time is bounded above by $\mathcal{O}(\binom{L}{k}k2^k)$, this is quite practical when compared to the alternative provided by a straight Fast Walsh transform of $\mathcal{O}(L2^L)$. For example, if $k = 3$ then the execution time for the embedded landscape approach is $\mathcal{O}(L^3)$.

In the case of the more general Kargupta's algorithm for a function that is epistatically bounded by k, the number of nonzero Walsh coefficients is $\mathcal{O}\binom{L}{k}$. All of the k-order Walsh coefficients can be computed by averaging 2^k function evaluations and Walsh function evaluations (assuming random Walsh coefficients) for each of the $\binom{L}{k}$ coefficients. The $(k-1)$-order Walsh coefficients can now be computed using 2^{k-1} function evaluations and subtracting away the effects of the k-order Walsh coefficients. This process can certainly be done in $\mathcal{O}(kL\binom{L}{k}2^k)$ operations, if k is known in advance.

Given that the nonzero Walsh coefficients are now identified in both classes of functions, Theorem 3 can clearly be used to compute the r^{th} moment in $\mathcal{O}(n^r)$ time, where n is the number of nonzero Walsh coefficients. Since n is polynomial in size relative to L so is n^r for fixed r independent of L. Since both the Walsh coefficient calculation and the moment computation can be carried out in polynomial time for fixed r, any summary statistics can be computed from Theorem 3 in polynomial time.

9 Hyperplane Statistics

A similar approach to that used in the last section can be used to calculate summary statistics for a given hyperplane. Hyperplane statistics can be used to study the distribution of hyperplane fitnesses and make statistical inferences about the effectiveness of hyperplane sampling. There are two types of moments for a hyperplane: the moment about the mean of the entire function and the moment about the mean of just the hyperplane itself. We will treat these two cases in that order.

THEOREM 4 (Moment of Hyperplane about the Function Mean): *The r^{th} moment of the elements of hyperplane h about the mean μ for function f given the Walsh coefficients of f is:*

$$\mu_r(h) = \sum_{a_1 \oplus \ldots \oplus a_r \subseteq \alpha(h)} w_{a_1} \ldots w_{a_r} \psi_{a_1 \oplus \ldots \oplus a_r}(\beta(h)),\ a_i \neq 0 \,\forall\, i$$

PROOF: From the definition of r^{th} moment and assuming an equal probability of selecting any domain value in the hyperplane:

$$\mu_r(h) = \frac{1}{|h|} \sum_{x \in h} (f(x) - \mu)^r$$

where μ is the mean for the entire function. We now proceed as with the earlier derivation:

$$\mu_r(h) = \frac{1}{|h|} \sum_{a_1=1}^{2^L-1} \sum_{a_2=1}^{2^L-1} \cdots \sum_{a_r=1}^{2^L-1} w_{a_1} w_{a_2} \ldots w_{a_r} \sum_{x \in h} \psi_{a_1 \oplus a_2 \oplus .. \oplus a_r}(x)$$

Using the fact that (Heckendorn and Whitley, 1999)

$$\sum_{x \in h} \psi_j(x) = \begin{cases} 0 & \text{if } j \not\subseteq \alpha(h) \\ \psi_j(\beta(h))|h| & \text{if } j \subseteq \alpha(h) \end{cases}$$

we get:

$$|h|\mu_r(h) = \sum_{a_1 \oplus a_2 \oplus \ldots \oplus a_r \subseteq \alpha(h)} w_{a_1} \ldots w_{a_r} (\psi_{a_1 \oplus \ldots \oplus a_r}(\beta(h))|h|), \quad \text{where} \quad a_i \neq 0 \,\forall\, i$$

Therefore, the r^{th} moment about the mean for the entire function over hyperplane h is:

$$\mu_r(h) = \sum_{a_1 \oplus a_2 \oplus \ldots \oplus a_r \subseteq \alpha(h)} w_{a_1} w_{a_2} \ldots w_{a_r} \psi_{a_1 \oplus a_2 \oplus \ldots \oplus a_r}(\beta(h)), \quad \text{where} \quad a_i \neq 0 \,\forall\, i$$

□

Now consider the case where the mean used in the moment calculations is the mean of the hyperplane. We denote this moment for hyperplane h about the mean of h as $\widehat{\mu}_r(h)$.

THEOREM 5 (Moment of Hyperplane about Hyperplane Mean): *The moment of the elements of hyperplane h about the mean, $\widehat{\mu}$, for hyperplane h in terms of the Walsh coefficients of f is:*

$$\widehat{\mu}_r(h) = \sum_{a_1 \oplus a_2 \oplus \ldots \oplus a_r \subseteq \alpha(h)} w_{a_1} w_{a_2} \ldots w_{a_r} \psi_{a_1 \oplus a_2 \oplus \ldots \oplus a_r}(\beta(h)), \quad \textit{where} \quad a_i \not\subseteq \alpha(h) \,\forall\, i$$

PROOF: Returning to the original equation for moment we get:

$$\widehat{\mu}_r(h) = \frac{1}{|h|} \sum_{x \in h} (f(x) - \widehat{\mu})^r$$

The Hyperplane Averaging theorem (Heckendorn and Whitley, 1999) states:

$$\frac{1}{|h|} \sum_{x \in h} f(x) = \sum_{j \subseteq \alpha(h)} w_j \psi_j(\beta(h))$$

Substituting the Walsh transform for the function f and using the Hyperplane Averaging theorem for $\widehat{\mu}$ we get:

$$\widehat{\mu}_r(h) = \frac{1}{|h|} \sum_{x \in h} \left(\sum_{i=0}^{2^L-1} w_i \psi_i(x) - \sum_{k \subseteq \alpha(h)} w_k \psi_k(\beta(h)) \right)^r$$

The left sum in parentheses can now be broken into two parts

$$\widehat{\mu}_r(h) = \frac{1}{|h|} \sum_{x \in h} \left(\sum_{i \subseteq \alpha(h)} w_i \psi_i(x) + \sum_{j \not\subseteq \alpha(h)} w_j \psi_j(x) - \sum_{k \subseteq \alpha(h)} w_k \psi_k(\beta(h)) \right)^r$$

Regrouping under the sums gives

$$\widehat{\mu}_r(h) = \frac{1}{|h|} \sum_{x \in h} \left(\sum_{i \subseteq \alpha(h)} \left(w_i \psi_i(x) - w_i \psi_i(\beta(h)) \right) + \sum_{j \not\subseteq \alpha(h)} w_j \psi_j(x) \right)^r$$

Note that $x \in h$ and $i \subseteq \alpha(h)$, therefore, $\psi_i(x) = \psi_i(\beta(h))$! This means $w_i \psi_i(x) - w_i \psi_i(\beta(h))$ is zero and we get

$$\widehat{\mu}_r(h) = \frac{1}{|h|} \sum_{x \in h} \left(\sum_{j \not\subseteq \alpha(h)} w_j \psi_j(x) \right)^r$$

Proceeding with the expansion of the r^{th} power as we did in the earlier proofs:

$$\begin{aligned}
\widehat{\mu}_r(h) &= \frac{1}{|h|} \sum_{x \in h} \sum_{a_1 \not\subseteq \alpha(h)} \sum_{a_2 \not\subseteq \alpha(h)} \cdots \sum_{a_r \not\subseteq \alpha(h)} w_{a_1} w_{a_2} \ldots w_{a_r} \psi_{a_1 \oplus a_2 \oplus \ldots \oplus a_r}(x) \\
&= \frac{1}{|h|} \sum_{a_1 \not\subseteq \alpha(h)} \sum_{a_2 \not\subseteq \alpha(h)} \cdots \sum_{a_r \not\subseteq \alpha(h)} w_{a_1} w_{a_2} \ldots w_{a_r} \sum_{x \in h} \psi_{a_1 \oplus a_2 \oplus \ldots \oplus a_r}(x) \\
&= \frac{1}{|h|} \sum_{a_1 \not\subseteq \alpha(h)} \sum_{a_2 \not\subseteq \alpha(h)} \cdots \sum_{a_r \not\subseteq \alpha(h)} w_{a_1} w_{a_2} \ldots w_{a_r} |h| \psi_{a_1 \oplus a_2 \oplus \ldots \oplus a_r}(\beta(h)), \\
&\qquad \text{with } a_1 \oplus a_2 \oplus \ldots \oplus a_r \subseteq \alpha(h) \\
&= \sum_{a_1 \not\subseteq \alpha(h)} \sum_{a_2 \not\subseteq \alpha(h)} \cdots \sum_{a_r \not\subseteq \alpha(h)} w_{a_1} w_{a_2} \ldots w_{a_r} \psi_{a_1 \oplus a_2 \oplus \ldots \oplus a_r}(\beta(h)), \\
&\qquad \text{with } a_1 \oplus a_2 \oplus \ldots \oplus a_r \subseteq \alpha(h)
\end{aligned}$$

Which is the same as saying:

$$\widehat{\mu}_r(h) = \sum_{a_1 \oplus a_2 \oplus \ldots \oplus a_r \subseteq \alpha(h)} w_{a_1} w_{a_2} \ldots w_{a_r} \psi_{a_1 \oplus a_2 \oplus \ldots \oplus a_r}(\beta(h)), \qquad \text{where } a_i \not\subseteq \alpha(h) \; \forall \, i$$

□

Note that when h is fixed as all *'s, that is, h is the whole domain then, $\alpha(h) = 0$ and the Moment of Hyperplane about Hyperplane Mean Theorem becomes the same as the first theorem in this paper.

The execution time for the statistics from the last two theorems is again polynomial in the number of nonzero Walsh coefficients. Therefore, the total execution time to compute the statistic for an embedded landscape is polynomial in the number of bits in the domain. The actual selection of the a_i's makes the computation $\mathcal{O}(n^r)$ where n is the number of nonzero Walsh coefficients.

9.1 Statistics by Partition

It is hyperplane competition within each partition and the ranking (Heckendorn et al., 1997) of the supporting hyperplanes in a partition that significantly influences the direction of convergence of a GA.

For a given partition, all of the hyperplanes in the partition will have the same α but they will all have unique a β. This means for any hyperplane in a fixed partition the hyperplane statistics for that hyperplane are computed by summing over the same set of a_i. Only the values $\beta(h)$ change. Or said another way, the difference in the hyperplane statistics for two hyperplanes h_1 and h_2, both from the same partition is the sign of the products of the Walsh coefficients that are summed. The sign being determined by:

$$\psi_{a_1 \oplus a_2 \oplus \ldots \oplus a_r}(\beta(h_i))$$

This means that the hyperplane statistics for all hyperplanes in a partition can be quickly computed without any extra multiplication of Walsh coefficients or deciding which indices a_i to sum over. This allows us to quickly compute comparative statistics between competing hyperplanes in a given partition.

10 Conclusions

In this paper we have introduced embedded landscapes as an extension of NK landscapes and MAXSAT problems. This extension is valid for any problem where the representation can be expressed as a sum of subfunctions over subsets of the representation domain. This would encompass many additive constraint problems and problems expressed as the interaction of subcomponents where the critical features of the subcomponents are represented by subsets of bits in the domain.

We show that embedded landscapes of fixed maximum epistasis K is exponentially sparse in epistatic space with respect to all possible functions. This gives us a handle with which to compute in polynomial time many features about these functions including all the epistatic interactions via Walsh analysis, and statistical moments of hyperplanes about the function mean and hyperplane mean. These moments give us comparative statistics about a polynomial number of hyperplanes in polynomial time. We also showed that embedded landscapes of even small fixed K can be NP-complete. We can conclude that with all of this epistatic and hyperplane data for any embedded landscape of fixed maximal epistasis computed in polynomial time, the difficulty of the

problem lies not in the epistasis itself but in the interaction of the epistatic parts. That is, knowing the epistasis and hyperplane statistics is not enough to solve the exponentially difficult part of these general problems.

Acknowledgments

This material is based upon work supported by the National Science Foundation under grant no. 9503366. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

Altenberg, L. (1994). Evolving better representations through selective genome growth. In *Proceedings of the IEEE World Congress on Computational Intelligence*, pages 182–187, IEEE Press, Piscataway, New Jersey.

Altenberg, L. (1996). Nk fitness landscapes. In Bäck, T., Fogel, D., and Michalewicz, Z., editors, *The Handbook of Evolutionary Computation*, Oxford University Press, Oxford, UK.

Bethke, A. D. (1981). *Genetic Algorithms as Function Optimizers*. Ph.D. thesis, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, Michigan.

Cormen, T. H., Leiserson, C. E., and Rivest, R. L. (1990). *Introduction to Algorithms*. McGraw-Hill, New York, New York.

Goldberg, D. (1989a). Genetic algorithms and walsh functions: Part i, a gentle introduction. *Complex Systems*, 3:129–152.

Goldberg, D. (1989b). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, Massachusetts.

Heckendorn, R. B. and Whitley, D. (1997). A walsh analysis of nk-landscapes. In Bäck, T., editor, *Proceedings of the Seventh International Conference on Genetic Algorithms*, pages 41–48, Morgan Kaufmann, San Francisco, California.

Heckendorn, R. B. and Whitley, D. (1999). Predicting epistasis from mathematical models. *Evolutionary Computation*, 7(1):69–101.

Heckendorn, R. B., Rana, S., and Whitley, D. (1999a). Polynomial time summary statistics for a generalization of maxsat. In Banzhaf, W. et al., editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 1, pages 281–288, Morgan Kaufmann, San Francisco, California.

Heckendorn, R. B., Rana, S., and Whitley, D. (1999b). Test function generators as embedded landscapes. In Bäck, T. and Banzhaf, W., editors, *Foundations of Genetic Algorithms - 5*, pages 183–198, Morgan Kaufmann, San Francisco, California.

Heckendorn, R. B., Whitley, L. D., and Rana, S. (1997). Nonlinearity, hyperplane ranking and the simple genetic algorithm. In Belew, R. K. and Vose, M., editors, *Foundations of Genetic Algorithms - 4*, pages 181–202, Morgan Kaufmann, San Francisco, California.

Hogg, T., Huberman, B. A., and Williams, C. P. (1996). Special issue sat problems. *Artificial Intelligence*, 81(1-2).

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

Jones, T. (1995a). *Evolutionary Algorithms, Fitness Landscapes and Search*. Ph.D. thesis, University of New Mexico, Albuquerque, New Mexico.

Jones, T. (1995b). One operator, one landscape. Technical Report SFI TR 95-02-025, Santa Fe Institute, Santa Fe, New Mexico.

De Jong, K. A., Potter, M. A., and Spears, W. M. (1997). Using problem generators to explore the effects of epistasis. In Bäck, T., editor, *Proceedings of the Seventh International Conference on Genetic Algorithms*, pages 338–339, Morgan Kaufmann, San Francisco, California.

Kargupta, H. and Park, B. H. (1999). Fast construction of distributed and decomposed evolutionary representation. In Brave, S. and Wu, A. S., editors, *Late Breaking Papers at the 1999 Genetic and Evolutionary Computation Conference*, pages 139–148, Morgan Kaufmann, San Francisco, California.

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

Kauffman, S. A. (1995). *At Home in the Universe*. Oxford University Press, Oxford, UK.

Mendenhall, W. (1967). *Introduction to Probability and Statistics*. Second Edition. Wadsworth Publishing, Belmont, California.

Niven, I. (1965). *Mathematics of Choice*. Mathematical Association of America, Washington, DC.

Papadimitriou, C. H. (1994). *Computational Complexity*. Addison-Wesley, Reading, Massachusetts.

Reeves, C. and Wright, C. (1995). An experimental design perspective on genetic algorithms. In Whitley, D. and Vose, M., editors, *Foundations of Genetic Algorithms - 3*, pages 7–22, Morgan Kaufmann, San Francisco, California.

Solé, R. and Goodman, B. (2000). *Signs of Life: How Complexity Pervades Biology*. Basic Books, New York, New York.

Weinberger, E. (1990). Correlated and uncorrelated fitness landscapes and how to tell the difference. *Biological Cybernetics*, 63:325–226.

Weinberger, E. (1996). Np completeness of kauffman's n-k model, a tuneably rugged fitness landscape. `www.santafe.edu/sfi/publications/Working-Papers/96-02-003.ps`.

Whitley, D., Mathias, K., and Pyeatt, L. (1995). Hyperplane ranking in simple genetic algorithms. In Eshelman, L. J., editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 231–238, Morgan Kaufmann, San Francisco, California.