B2.6.1 Transform methods: Walsh analysis, Walsh transforms

Sami Khuri
Department of Mathematics and Computer Science
San José State University
One Washington Square
San José, CA 95192-0103, USA
khuri@sjsumcs.sjsu.edu
Fax: (408)924-5080

Abstract

Theoretical analysis of fitness functions in genetic algorithms has included the use of Walsh functions. They form a convenient basis for the expansion of fitness functions. These orthogonal, rectangular functions, which take values of ± 1, are more practical as a basis than the traditional sine/cosine basis. Walsh functions have also been used to compute the average fitness values of schemata, to decide whether a certain function is hard or easy for a genetic algorithm, and to design deceptive functions for the genetic algorithm.

Introduction

Traditionally, Fourier series and transforms have been used to represent large classes of functions by superpositioning sine and cosine functions. More recently, other classes of complete, orthogonal functions are being used for the same purpose (Harmuth 1968). These new functions are rectangular and are easier to define and use with digital logic. Walsh functions have been used for analyzing various natural events. Much of the information pertaining to such events is in the form of signals which are functions of time (Beauchamp 1984). These signals can be studied, classified and analyzed using orthogonal functions and transformations. Walsh functions form a set whose members are simple functions which are easy to generate and define. Discrete and continuous signals can be represented as combinations of members of Walsh functions. Only orthogonal functions can completely synthesize a given time function accurately. Orthogonal functions also possess many important mathematical properties which makes them highly useful as an analysis tool.

Walsh functions form a complete orthogonal set of functions and can thus be used as a basis. These rectangular functions, which take values ± 1, are more practical, as a basis, than the traditional trigonometric basis (Bethke 1980). They have been used as theoretical tools to compute the average fitness values of hyperplanes, to decide whether a certain function is hard or easy for a genetic algorithm, and to design deceptive functions for the genetic algorithm (Bethke 1980, Goldberg 1989a, 1989b).

In this section, most of which is from Golberg's articles (Goldberg 1989a, 1989b), we explore the use of Walsh functions as bases for fitness functions of genetic algorithms. In the second section, B2.6.2, an alternative to Walsh functions, Haar functions, are investigated, followed by a comparison between the two.

Walsh functions as bases

Walsh transforms are orthogonal, rectangular functions which are very practical as bases for fitness functions of genetic algorithms (Bethke 1980).

Walsh functions form an ordered set of rectangular waveforms taking one of two amplitude values $+1$ and -1. They form a complete orthogonal set of functions and can thus be used as a basis (Bethke 1980). Consequently, any fitness function defined on binary strings of length ℓ can be represented as a linear combination of discrete Walsh functions.

When working with Walsh functions, it is more convenient to have strings with ± 1 rather than 0 and 1. The *auxilliary function* (Goldberg 1989a) associates with each binary string $x = x_\ell x_{\ell-1} \ldots x_2 x_1$ an auxilliary string $y = y_\ell y_{\ell-1} \ldots y_2 y_1$ where $y_i = 1 - 2x_i$ for $i = 1, 2, \ldots, \ell$ or equivalently $x_i = \frac{1}{2}(1 - y_i)$. Hence $y_i \in \{+1, -1\}$ for all $i = 1, \ldots, \ell$. Given a string x, the auxilliary string y is defined as $y = aux(x_\ell)aux(x_{\ell-1}) \ldots aux(x_2)aux(x_1)$ where

$$y_i = aux(x_i) = \begin{cases} -1 & \text{if } x_i = 1 \\ 1 & \text{if } x_i = 0 \end{cases}$$

The *Walsh functions* (monomials (Goldberg 1989a)) over auxilliary string variables form a set of 2^ℓ monomials defined for $y = aux(x)$:

$$\Psi_j(y) = \Pi_{i=1}^{\ell} y_i^{j_i}$$

where $j = j_\ell j_{\ell-1} \ldots j_1$ and $j = \Sigma_{i=1}^{\ell} j_i 2^{i-1}$.

Example Let us compute the monomial $\Psi_3(y)$ where $\ell = 3$. Since $j = 3$, $j_3 = 0$, $j_2 = 1$ and $j_1 = 1$. Therefore $\Psi_3(y) = y_3^0 y_2^1 y_1^1 = y_1 y_2$. In other words, $j = 3$ signals the presence of positions one and two, which correspond to the indices of y ($\Psi_3(y) = y_1 y_2$). This ease of conversion between the indices and the Walsh monomials is one of the reasons behind Walsh's success as a basis. If $x = 101$ then we may write

$$\begin{aligned}\Psi_3(aux(101)) &= \Psi_3(aux(1)aux(0)aux(1)) \\ &= \Psi_3(-11-1) \\ &= (-1)(1) \\ &= -1\end{aligned}$$

The set $\{\Psi_j(y) \ : \ j = 0, 1, 2, \ldots, 2^\ell - 1\}$ forms a basis for the fitness functions defined on $[0, 2^\ell)$. That is:

$$f(x) = \Sigma_{j=0}^{2^\ell - 1} w_j \Psi_j(x), \tag{B2.6.1}$$

where the w_j 's are the Walsh coefficients, $\Psi_j(x)$ are the Walsh monomials, and by $\Psi_j(x)$ we mean $\Psi_j(aux(x))$. The Walsh coefficients are given by:

$$w_j = \frac{1}{2^\ell} \Sigma_{x=0}^{2^\ell - 1} f(x) \Psi_j(x). \tag{B2.6.2}$$

The subscript j of the Walsh monomial Ψ_j denotes the index. The number of 1's in the binary expansion of j is the weight of the index. Thus, $\Psi_3(x)$ has 3 as index and is of weight 2.

Since $\Psi_j(x) \neq 0$ for each $j \in [0, 2^\ell)$, unless $f(x)$ is orthogonal to a Walsh function $\Psi_j(x)$, the expansion of $f(x)$ as a linear combination of Walsh monomials has 2^ℓ non-zero terms. Thus, at most 2^ℓ non-zero terms are required for the expansion of a given function as a linear combination of Walsh functions. The total number of terms required for the computation of the expansion of the fitness function at a given point is $2^{2\ell}$.

$\Psi_j(x)$ is defined for discrete values of $x \in [0, 2^\ell)$. The function can be extended to obtain step functions (Beauchamp 1984) by allowing all values of t in the interval $[0, 2^\ell)$ and letting: $\Psi_j(t) = \Psi_j(x_s)$ for $t \in [x_s, x_s + 1)$.

Table 1 gives the Walsh functions on $[0, 8)$.

x	$\Psi_0(x)$	$\Psi_1(x)$	$\Psi_2(x)$	$\Psi_3(x)$	$\Psi_4(x)$	$\Psi_5(x)$	$\Psi_6(x)$	$\Psi_7(x)$
000	1	1	1	1	1	1	1	1
001	1	-1	1	-1	1	-1	1	-1
010	1	1	-1	-1	1	1	-1	-1
011	1	-1	-1	1	1	-1	-1	1
100	1	1	1	1	-1	-1	-1	-1
101	1	-1	1	-1	-1	1	-1	1
110	1	1	-1	-1	-1	-1	1	1
111	1	-1	-1	1	-1	1	1	-1
index j	0	1	2	3	4	5	6	7
$j_3 j_2 j_1$	000	001	010	011	100	101	110	111
monomials	$y_3^0 y_2^0 y_1^0$	$y_3^0 y_2^0 y_1^1$	$y_3^0 y_2^1 y_1^0$	$y_3^0 y_2^1 y_1^1$	$y_3^1 y_2^0 y_1^0$	$y_3^1 y_2^0 y_1^1$	$y_3^1 y_2^1 y_1^0$	$y_3^1 y_2^1 y_1^1$
$y_3^{j_3} y_2^{j_2} y_1^{j_1}$	1	y_1	y_2	$y_1 y_2$	y_3	$y_1 y_3$	$y_2 y_3$	$y_1 y_2 y_3$

Table 1. Walsh functions on $[0, 8)$.

The Walsh monomials presented here are in natural order, also known as Hadamard order. Other well-known orderings include the sequency order (Walsh 1923), the Rademacher order (Alexits 1961), the Harmuth order (Harmuth 1972), and the boolean synthesis order (Gibbs 1970).

The natural order is known as Hadamard because the Walsh coefficients can be obtained using the Hadamard matrix. The rows and columns of the Hadamard matrix are orthogonal to one another. The lowest-order Hadamard matrix is $H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$. The following recursive relation generates higher-order Hadamard matrices of order N:

$$H_N = H_{\frac{N}{2}} \otimes H_2$$

where $\otimes$ is the Kronecker or direct product. The Kronecker product in the above equation consists in replacing each element in H_2 by $H_{\frac{N}{2}}$; that is, 1's are replaced by H_2 and -1's by $-H_2$. This is called the Sylvester construction (MacWilliams 1977) and thus:

$$H_N = \begin{bmatrix} H_{\frac{N}{2}} & H_{\frac{N}{2}} \\ H_{\frac{N}{2}} & -H_{\frac{N}{2}} \end{bmatrix}$$

For instance, H_8 is the following matrix:

$$H_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}$$

The Walsh coefficients can be obtained by the following equality:

$$\begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{2^\ell - 1} \end{bmatrix} = \frac{1}{2^\ell} H_{2^\ell} \begin{bmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_{2^\ell - 1}) \end{bmatrix}$$

The second reason behind Walsh functions' popularity in genetic algorithms is the relatively small number of terms required in the expansion of schmeta as linear combinations of Walsh coefficients.

Schema average fitness

The average fitness value of a schema can be expressed as a partial signed summation over Walsh coefficients. Recall that a schema h (or hyperplane in the search space) is of the form: $h_\ell, h_{\ell-1}, \ldots, h_2, h_1$ where $h_i \in \{0, 1, \star\}$ for all $i = 1, \ldots, \ell$, where $\star$ is either 0 or 1.

The schema average fitness values can be written in terms of Walsh coefficients. If $|h|$ denotes the number of elements in schema h, then the schema average fitness, $f(h)$, is given by (Bethke 1980) and (Goldberg 1989a):

$$f(h) = \frac{1}{|h|} \Sigma_{x \in h} f(x).$$

By substituting the value of $f(x)$, given by Equation B2.6.1, and by rearranging the summations, we obtain:

$$\begin{aligned} f(h) &= \frac{1}{|h|} \Sigma_{x \in h} \Sigma_{j=0}^{2^\ell - 1} w_j \Psi_j(x) \\ &= \frac{1}{|h|} \Sigma_{j=0}^{2^\ell - 1} w_j \Sigma_{x \in h} \Psi_j(x), \end{aligned}$$

where by $\Psi_j(x)$ we mean $\Psi_j(aux(x))$. Expressing the schema (rather than the individual points of the search space) as a linear combination of Walsh functions has a very advantageous effect: the lower the order of the schema, the fewer the number of terms in the summation. As can be seen in Table 2, in which six schemata of length 3 are written as linear combinations of the Walsh coefficients, the schemata with low order, such as $\star\star 0$, need only two terms, while 001 is of higher order and has eight terms in its expansion.

Schema	Fitness Averages
$\star\star\star$	w_1
$\star\star 0$	$w_0 + w_1$
$1\star\star$	$w_0 - w_4$
$\star 01$	$w_0 - w_1 + w_2 - w_3$
$11\star$	$w_0 - w_2 - w_4 + w_6$
001	$w_0 - w_1 + w_2 - w_3 + w_4 - w_5 + w_6 - w_7$

Table 2. Fitness averages as partial Walsh sums of some schemata.

Easy and hard problems for genetic algorithms

In his quest to define functions that are suitable for genetic algorithms, Goldberg, using Walsh coefficients, first designed minimal deceptive functions of two types, and later constructed fully deceptive 3-bit functions. If 111 is the optimal point in the search space, then all order-one schemata lead away from it. In other words, we have $f(\star\star 1) < f(\star\star 0)$, $f(\star 1\star) < f(\star 0\star)$, and $f(1\star\star) < f(0\star\star)$.

Similarly, for order-two schemata, we want $f(\star 00) > f(\star 01)$, $f(\star 00) > f(\star 10)$, and $f(\star 00) > f(\star 11)$. By using the methods described in the previous section, the above inequalities are cast in terms of Walsh coefficients, and the problem reduces to solving a simultaneous set of inequlities. Many such fully deceptive functions can be designed.

Goldberg (Goldberg 1989b) also used neighborhood arguments and devised ANODE: *an*alysis *o*f *de*ception an algorithm for determining whether and to what degree a coding-function combination is deceptive. The algorithm is applied to an instance of a problem, and decides whether or not that particular instance is deceptive for the genetic algorithm. In other words, the diagnostic is problem-instance dependent, unlike other, more general theories that associate labels with problems (and not instances), such as tractability issues and the classification of problems into the classes of P and NP-complete, for instance.

It is also crucial to keep in mind that the above analysis is of static nature, and that simple genetic algorithms were not designed to be optimization algorithms for static optimization problems in the first place (DeJong 1992). For a different kind of analysis with a more dynamic flavor, the reader is referred to Fogel's (Fogel 1992) and Rudolph's (Rudolph 1994) work, for instance, in which Markov chains are used to model genetic algorithms and to tackle conditions under which global convergence is achieved.

Fast Walsh transforms

By taking advantage of the many repetitive computations performed with orthogonal transformations, the analysis can be implemented in the order of at most $N \log_2 N$ computations to form *fast transforms*. Note that $N = 2^\ell$. Modelled after the fast Fourier transform (Cooley and Tukey 1965), several fast Walsh transforms have been proposed (Shanks 1969). Since memory storage for intermediate-stage computations is not needed, these are in-place algorithms The calculated pair of output values can replace the corresponding pair of data in the preceding stage.

Conclusion

In summary, Walsh functions form convenient bases. They can thus be used as practical transforms for discrete objective functions in optimization problems such as fitness functions in genetic algorithms. They are used to calculate the average fitness value of a schema, to decide whether a certain function is hard or easy for a genetic algorithm, and to design deceptive functions for the genetic algorithm.

References

Alexits G 1961 *Convergence Problems of Orthogonal Series* (New York and London: Pergammon)

Beauchamp K G 1984 *Applications of Walsh and Related Functions* (Academic Press)

Bethke A D 1980 *Genetic Algorithms as Function Optimizers* Doctoral dissertation University of Michigan

Cooley J W and Tukey J W 1965 An Algorithm for the Machine Calculation of Complex Fourier Series *Math. Computation* **19** 297–301

De Jong K A 1992 Are Genetic Algorithms Function Optimizers? *Parallel Problem Solving from Nature* **2** 3–13 (North-Holland)

Fogel D B 1992 *Evolving Artificial Intelligence* Doctoral dissertation University of California at San Diego

Gibbs J E 1970 Discrete Complex Walsh Transforms *Proc. of the Symposium on Applications of Walsh Functions* (Washington, D.C.) 106–122

Goldberg D E 1989a Genetic Algorithms and Walsh Functions: Part I, A Gentle Introduction *Complex Systems* **3** 129–52

Goldberg D E 1989b Genetic Algorithms and Walsh Functions Part II: Deception and its Analysis *Complex Systems* **3** 153–71

Haar A 1910 Zur Theorie der Orthogonalen Funktionensysteme *Math. Annalen* **69** 331–71

Harmuth H F 1968 A Generalized Concept of Frequency and Some Applications *IEEE Transactions on Information Theory* **14** 375–82

Harmuth H F 1972 *Transmission of Information by Orthogonal Functions* (Berlin: Springer-Verlag, second edition)

Karpovsky M G 1985 *Spectral Techniques and Fault Detection* (Academic Press)

Kremer H 1973 On Theory of Fast Walsh Transform Algorithms *Colloquium on the Theory and Applications of Walsh and Other Non-Sinusoidal Functions (Hatfield, U.K.)*

MacWilliams F J and Sloane N J A 1977 *The Theory of Error-Correcting Codes* (North-Holland, New York)

Nagy B S 1965 *Introduction to Real Functions and Orthogonal Expansions* (New York: Oxford University Press) 338–340

Roeser P R and Jernigan M E 1982 Fast Haar Transform Algorithms *IEEE Transactions on Computers* **31** 175–7

Rudolph G 1994 Convergence Analysis of Canonical Genetic Algorithms *IEEE Transactions on Neural Networks* **31** 96–101

Shanks J L 1969 Computation of the Fast Walsh-Fourier Transform *IEEE Transactions on Computers* **5** 96–101

Walsh J L 1923 A Closed Set of Orthogonal Functions *Ann. J. Math* **55** 5–24

