A Methodology for the Statistical Characterization of Genetic Algorithms

Angel Fernando Kuri-Morales[1]

[1] Instituto Tecnológico Autónomo de México
Río Hondo No. 1, México D.F.
akuri@rhon.itam.mx

Abstract. The inherent complexity of the Genetic Algorithms (GAs) has led to various theoretical an experimental approaches whose ultimate goal is to better understand the dynamics of such algorithms. Through such understanding, it is hoped, we will be able to improve their efficiency. Experiments, typically, explore the GA's behavior by testing them *versus* a set of functions with characteristics deemed adequate. In this paper we present a methodology which aims at achieving a solid relative evaluation of alternative GAs by resorting to statistical arguments. With it we may categorize any iterative optimization algorithm by statistically finding the basic parameters of the probability distribution of the GA's optimum values without resorting to *a priori* functions. We analyze the behavior of 6 algorithms (5 variations of a GA and a hill climber) which we characterize and compare. We make some remarks regarding the relation between statistical studies such as ours and the well known "No Free Lunch Theorem".

1 Introduction

The appeal of using GAs in optimization problems is largely dependent on the fact that they make very light demands on the characteristics of the functions to optimize. Their users, on the other hand, have to cope with the problem of choosing the right breed of GA. Theoretical studies [1], [2], [3], [4] shed some light on what to expect and [5], [6] and many others have tried to establish the reliability of a given GA experimentally. Here we focus on the experimental approach wherein, usually, conclusions are derived from selective and heuristically determined simulations. Rarely, if ever, these approaches may yield other than qualitative measures for two main reasons: a) The functions, usually determined "by hand" are limited in scope and range and b) The probability functions describing the algorithm's behavior are unknown and no bounds are, therefore, reachable. We propose a methodology where both problems are circumvented by a) Generating automatically the functions to optimize and b) Finding the parameters for the probability distributions of the best values from statistical theoretical considerations.

This paper is organized in 5 further sections. Section 2 describes the way in which unbiased functions are determined; section 3 succinctly describes the algorithms under study; section 4 describes the statistical method; in section 5 we present our results; finally, in section 6 we reach our conclusions and make some final remarks.

2 Automatic Generation of Unbiased Functions

To generate functions automatically we resort to Walsh functions $\psi_j(x)$ which form an orthogonal basis for real-valued functions defined on $(0,1)^l$, where x is a bit string and l is its length. By using such functions we allow for an easier cluster (schema) analysis of the results (not included in this paper). We restrict our study to the functions in $\Re^2$ and, hence, focus on functions of the form $y=f(x)$ but the method is easily extendible to $\Re^n$ [7]. Henceforth, any function F(x) thusly defined can be written as a linear combination of the ψ_j's (a Walsh polynomial).

$$F(x) = \sum_{j=0}^{2^l-1} \omega_j \psi_j(x) \tag{1}$$

where

$$\psi_j(x) = \begin{cases} +1 & if\ \pi(x \wedge j) = 0 \\ -1 & if\ \pi(x \wedge j) = 1 \end{cases} \tag{2}$$

$x \wedge j$ is the bitwise AND of x and j; $\pi(x)$ denotes the parity of x; and $\omega_j \in \Re$. Therefore, the index j and argument x of $\psi_j(x)$ must be expressed in comparable binary. We, therefore, used 48 bits to represent x in a fixed point format ± 23.24 (i.e. a sign bit; 23/24 bits for the integer/decimal parts of the number) and, consequently, also 48 bits for the index, i.e. $-2^{24} < x < +2^{24}$ and $0 \le j \le 2^{48}-1$. For example, a) $\psi_{267,386,880}(7) = -1$ or b) $\psi_{FF00000}(7.5) = +1$. We set a similar range for the Walsh coefficients, i.e. $-2^{24} < \omega_j < +2^{24}$. Therefore, any Walsh monomial $\omega_j \psi_j$ is uniquely represented by a binary string of length 96. Finally, we allow at least one but no more than 48 non-zero terms in (1). Given this last condition, (1) is replaced by

$$\gamma(x) = \sum_{j=1}^{48} \alpha_j \omega_j \psi_j(x) \tag{3}$$

where

$$\alpha_j = \begin{cases} 1 & if\ the\ j-th\ term\ is\ present \\ 0 & if\ the\ j-th\ term\ is\ not\ present \end{cases}$$

Denoting with τ the number of non-zero terms in 3 we see that a full (τ =48) function's binary representation is 4,608 bits long. We denote the space of all possible functions defined by (3) with Ξ and its cardinality with ξ. It is easy to see that $\xi \approx \sum_{i=1}^{47} (2^{96})^i$ which is a very large number. Therefore, the method outlined above provides us with an unlimited reservoir of functions in $\Re^2$. Equally

importantly, the random selection of a number τ and, thereafter, the further random selection of τ different indices and τ different ω_j's yields a uniquely identifiable function from such reservoir. It is also important to point out that, to make a fair comparison, a large (122,516) pool of Walsh functions was randomly generated. Then the $\gamma(x)$'s which the algorithms were required to minimize were all gotten from the same pool, thus allowing us to test the algorithms in a homogeneous functional environment.

3 Algorithms

We compared the following algorithms: a) A random mutation hill climber (RHC) described in [8], b) A simple (canonical) GA (CGA) described in [9] where, however, the best individual was externally preserved but did not participate in the genetic process, c) A simple eliTist (ETA) GA where the best individual was preserved and *did* participate in the genetic process, d) A statistical GA (SGA) described in [10] which does not rely on a population of individuals but, rather, on a unique statistical genome which captures the stochastic nature of the whole population, e) An eclectic GA (EGA) described in [11] which is actually a self-adaptive poli-algorithm (a GA with deterministic coupling/selection plus a RHC), f) A so-called Vasconcelos GA which is simply the EGA stripped of the RHC and self-adaptive mechanisms of the EGA. In what follows we refer to the algorithms in a) to f) as *A(i), i=1,...,6*. The interested reader may see the references.

It should be pointed out that, for the purposes of this work, we are restricting the use of these algorithms to minimize the functions of Ξ and that the process of minimization is unconstrained, i.e. we search for the least value of the functions in (3) in a pre-defined number of generations with the parameters illustrated in table 1. We use the following: $P_c \equiv$ probability of crossover, $P_m \equiv$ probability of mutation, $N \equiv$ population's size, $T \equiv$ size of elite; "*" means the parameter is self-adaptive, "-" means the parameter is not applicable.

Table 1. Operational Parameters for Selected Algorithms

Algorithm	P_c	P_m	N	T
CGA	0.9	0.005	50	1
TGA	0.9	0.005	50	1
EGA	*	*	*	50
SGA	-	0.005	50	5
VGA	0.9	0.005	50	50
RHC	-	-	1	1

4 Statistical Methodology

We want to answer the following. *Q*: For any given algorithm *A(i)*, what is the probability that we find a certain minimum value (denoted by κ) for any $\gamma(x)$ given that *A(i)* is iterated *G* times?

Since one of our premises is that the $\gamma(x)$ be selected randomly from Ξ we do not know, a priori, anything about the probability distribution function of the κ's. To answer *Q* we rely on the following known theorems from statistical theory.

T1) Any sampling distribution of means (sdom) is distributed normally for a large enough sample size *n*.

Remark: This is true, theoretically, as $n \to \infty$. However, it is considered that any n>20 is satisfactory. We have chosen n=36.

T2) In a normal distribution (with mean $\mu_{\bar{X}}$ and standard deviation $\sigma_{\bar{X}}$) approximately 1/10 of the observations lie in the intervals: $\mu_{\bar{X}} - 5\sigma_{\bar{X}}$ to $\mu_{\bar{X}}$ -1.29$\sigma_{\bar{X}}$; $\mu_{\bar{X}}$ -1.29$\sigma_{\bar{X}}$ to $\mu_{\bar{X}}$ –0.85$\sigma_{\bar{X}}$; $\mu_{\bar{X}}$ -0.85$\sigma_{\bar{X}}$ to $\mu_{\bar{X}}$ –0.53$\sigma_{\bar{X}}$; $\mu_{\bar{X}}$ -0.53$\sigma_{\bar{X}}$ to $\mu_{\bar{X}}$ – 0.26$\sigma_{\bar{X}}$; $\mu_{\bar{X}}$ -0.26$\sigma_{\bar{X}}$ to $\mu_{\bar{X}}$ and the symmetrical $\mu_{\bar{X}}$ to 0.26$\sigma_{\bar{X}}$, etc.

Remark: These deciles divide, therefore, the area under the normal curve in 10 unequally spaced intervals. The expected number of observed events in each interval will, however, be equal.

T3) The relation between the population distribution's parameters [which we denote with μ (the mean) and σ (the standard deviation)] and the sdom's parameters (which we denote with $\mu_{\bar{X}}$ and $\sigma_{\bar{X}}$) is given by $\mu = \mu_{\bar{X}}$ and $\sigma = \sqrt{n} \cdot \sigma_{\bar{X}}$.

Remark: In our case $\sigma = 6\sigma_{\bar{X}}$.

T4) The proportion of any distribution found within *k* standard deviations of the mean is, at least, $1\text{-}1/k^2$.

Remark: Chebyshev's bound generality makes it quite a loose one. Tighter bounds are achievable but they may depend on the characteristics of the distribution under study. We selected k = 4, which guarantees that our observations will occur with probability = 0.9375.

T5) For a set of *r* intervals, a number of O_i observed events in the *i*-th interval, a number of expected E_i events in the *i*-th interval, *p* distribution parameters and $\nu = r - p - 1$ degrees of freedom, the following equation holds.

$$P\left(Z \left| \sum_{i=1}^{r} \frac{(O_i - E_i)^2}{E_i} > c_0 + c_1\nu + c_2\nu^2 + c_3\nu^3 + c_4\nu^4 \right. \right) = 0.05 \qquad \textbf{(4)}$$

$c_0 \approx +1.98829512$

$c_1 \approx +2.06290867$

$c_2 \approx -0.06021040$

$c_3 \approx +0.00205163$

$c_4 \approx -0.00002637$

where $P(Z) \equiv$ probability that the distribution is normal.

Remarks: The summation on the left of (4) is the χ^2 statistic; the polynomial to the right of the inequality sign (call it $T(\nu)$) is a least squares Chebyshev polynomial approximation to the theoretical χ^2 for a 95% confidence level. In our case, $\nu = 7$ for which $T(\nu) \approx 14.0671$. Furthermore, if we choose the deciles as above, we know that $E_i = \eta/10 \; \forall i$, where η is a sample of size *n*. A further condition normally imposed on this goodness-of-fit test is that a minimum number of observations θ (usually between 3 and 5) be required in each interval. Thus, (4) is replaced by

$$P\left[Z \left| \left(\sum_{i=1}^{r} \frac{(O_i - E_i)^2}{E_i} > c_0 + c_1\nu + c_2\nu^2 + c_3\nu^3 + c_4\nu^4 \right) \vee (O_i < \theta \quad \forall i) \right. \right] = 0.05 \tag{5}$$

Making $\theta = 5$ and using the parameters' values described above, equation (5) finally takes the following form.

$$P\left[Z \left| \left(\sum_{i=1}^{10} \frac{100O_i^2 - 20\eta O_i + \eta^2}{10\eta} \le 14.0671 \right) \& (O_i \ge 5) \right. \right] = 0.95 \tag{6}$$

4.1 Algorithm for the Determination of the Distribution's Parameters

In what follows we describe the algorithm which is an evident conclusion resulting from all the foregoing considerations. We describe it for the characterization of any minimization algorithm. The reader should keep in mind that this is one of *A(i)*.

1. Generate a random binary string as per (3); this is one possible $\gamma(x)$.
2. Minimize $\gamma(x)$ iterating *A(i)* for *G* generations.
3. Store the best value κ.
4. Repeat steps (1-3) 36 times.
5. Calculate the average best value $\bar{\kappa}$.
6. Repeat steps (4-5) 50 times.
7. Calculate $\mu_{\bar{\kappa}}$ and $\sigma_{\bar{\kappa}}$.
8. Standardize the $\bar{\kappa}$'s.
9. Repeat steps 4-5,7-8 until χ^2 <14.0671 and $O_i \ge 5$.
10. The sdom's distribution is now known to be normal with *P(Z)=0.95*.
11. Calculate $\mu = \mu_{\bar{\kappa}}$ and $\sigma = 6\sigma_{\bar{\kappa}}$. We have extracted the expected best value of κ for this algorithm; we also know κ distribution's standard deviation.
12. In the absence of knowledge of the characteristics κ's probability distribution function we appeal to T4, from which we find:

$$P(\mu - 4\sigma \le \kappa \le \mu + 4\sigma) = 0.9375 \tag{7}$$

But now we can answer *Q*, for we know that $P(\kappa > \kappa) \leq 0.0625$ (where $\kappa = \mu + 4\sigma$: the worst case κ) or, equivalently, that $P(\kappa > \mu_{\kappa} + 24\sigma_{\kappa}) \leq 0.0625$. We now know that the probability that the best (minimum) value found by *A(i)* when minimizing $\gamma(x)$ (for any given *x* in Ξ) exceeds κ is statistically negligible. In other words, we have found a quantitative, unbiased measure of *A(i)*'s performance in $\Re^2$.

5 Results

The methodology just described was applied to all *A(i)* algorithms systematically increasing the number of generations *G(i)* such that *G(i)*=30,50,100,150 for *i*=1,2,3,4. We show the results for G(1-4) in tables 2-5. The column with heading "Relative" shows the performance relative to the best algorithm; "Samples" shows the number of κ 's that were needed to calculate before normality was achieved; "Funcs" denotes the number of functions which were minimized during this experiment.

Table 2. Comparison of Algorithms for 30 Generations

	μ_{κ}	σ_{κ}	κ	Relative	Samples	Funcs
VGA	-68.817	4.862	47.871	1.000	86	3096
EGA	-64.585	4.717	48.623	1.016	66	2376
TGA	-65.316	4.778	49.356	1.031	81	2916
CGA	-67.716	4.974	51.660	1.079	89	3204
SGA	-68.755	5.122	54.173	1.132	98	3528
RHC	-49.133	5.549	84.043	1.756	1170	3510

RHC was iterated so as to force comparable computational costs in all algorithms. The values in columns 2, 3 and 4 are, for convenience, divided by 10^6.

Table 3. Comparison of Algorithms for 50 Generations

	μ_{κ}	σ_{κ}	κ	Relative	Samples	Funcs
VGA	-67.212	4.505	40.908	1.000	72	2592
EGA	-68.271	4.778	46.401	1.134	72	2592
TGA	-70.618	5.012	49.670	1.214	71	2556
CGA	-69.668	5.081	52.276	1.278	87	3132
SGA	-70.796	5.380	58.324	1.426	63	2268
RHC	-49.133	5.549	84.043	2.054	70	3500

Table 4. Comparison of Algorithms for 100 Generations

	μ_{κ}	σ_{κ}	κ	Relative	Samples	Funcs

EGA	-68.780	4.855	47.740	1.000	70	2592
VGA	-73.295	5.123	49.657	1.040	70	2520
SGA	-73.947	5.202	50.901	1.066	70	2520
TGA	-69.768	5.138	53.544	1.122	66	2376
CGA	-71.713	5.381	57.431	1.203	86	3096
RHC	-49.181	5.529	83.515	1.749	35	3500

Table 5. Comparison of Algorithms for 150 Generations

	μ_κ	σ_κ	κ	Relative	Samples	Funcs
VGA	-74.535	4.020	21.945	1.000	76	2736
EGA	-69.065	3.828	22.810	1.039	69	2484
TGA	-71.813	4.120	27.067	1.233	73	2628
SGA	-77.519	4.360	27.121	1.236	77	2772
CGA	-73.788	4.233	27.810	1.267	87	3132
RHC	-49.133	5.549	84.041	3.830	24	3600

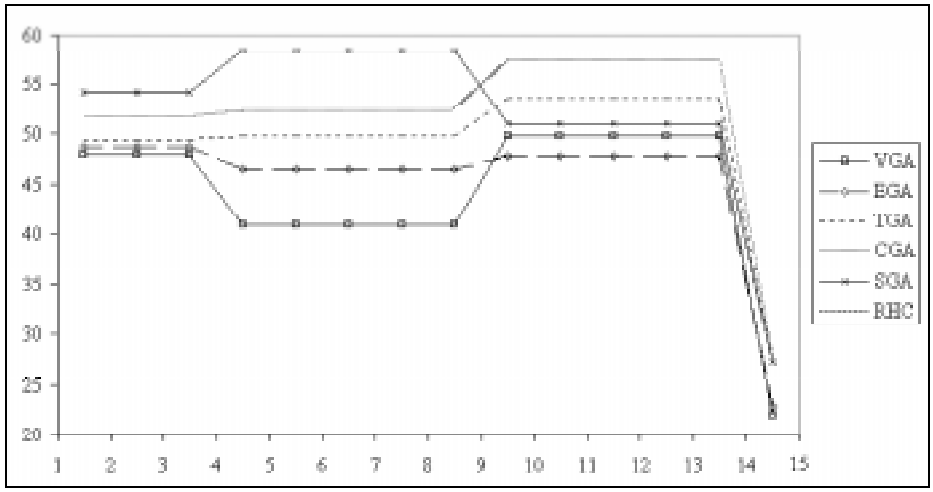
Notice that the number of samples needed was not homogeneous. This is interesting in that the typical method of finding the estimators for the mean and variance the size of the sample remains fixed. Here it is seen that a variable sample size is needed and that its size is considerable.

On the other hand, the best GA (relative to κ) does not remain constant, although in all cases VGA and EGA alternate the first and second places. Although EGA yields a poor κ for G(4), its low performance is offset by a lower (better) deviation. As expected, the RHC was worse than any of the GA variations.

When κ (and not κ) is considered, however, the order of performance is significantly modified. SGA displays the best κ throughout, with VGA improving as generations increase. EGA, on the other hand, displays a more modest behavior. Also notable is the fact that performance improves markedly when the number of generations is increased.

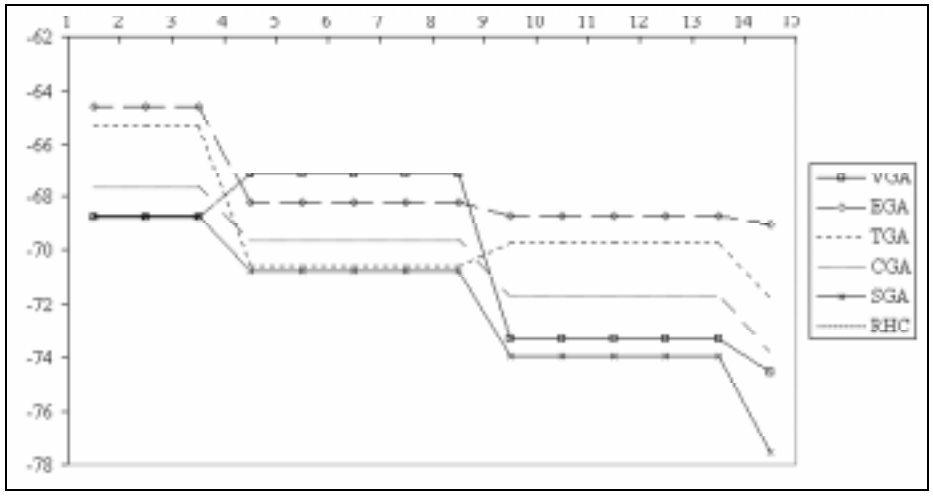
In the following three graphs we show the performances for κ's upper bound (κ), mean value (κ) and standard deviation (σ), respectively. The horizontal axis displays *G(i)*; its scale is divided by 10. The vertical axis displays κ, κ and σ, respectively; its scale is divided by 10^6.

Fig. 1. Performance of Different GAs for κ.



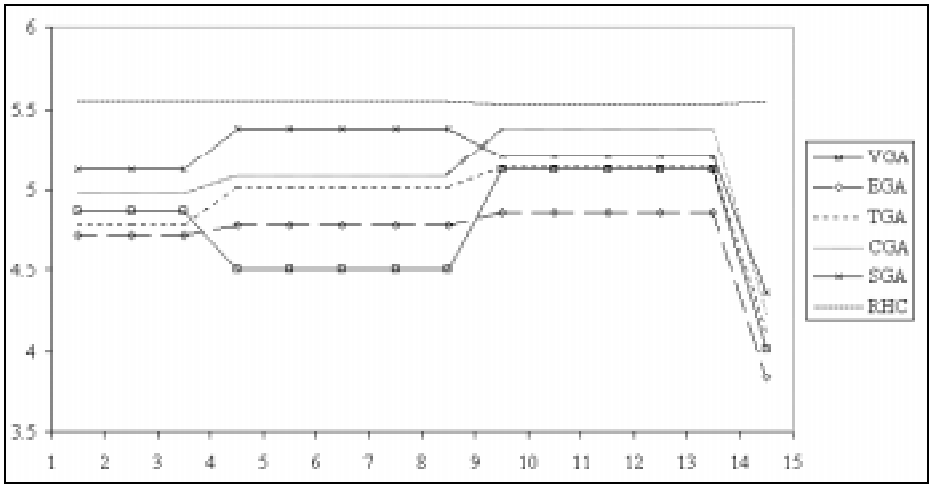
In figures 1 and 2 we have omitted the graph for RHC to allow for better reading of the other algorithm's performances.

Fig. 2. Performance of Different GAs for κ'.



In figure 3 we show how the standard deviations change with the number of generations. Of all algorithms EGA displays the most constant behavior. We may remark that this is so because of its self-adaptive nature.

Fig. 3. Standard Deviations for Different Algorithms



6 Conclusions

We have shown that it is possible to characterize a set of algorithms and quantify their expected behavior. To do this it is necessary to invest a considerable amount of computer resources. In the experiments reported, a total of 69,154 functions were minimized. In exchange for this considerable effort we extracted solid values for κ and κ.

Analysis of the resulting data may be extended well beyond the comments above. We are unable to do this here for reasons of space, but hope to do this in a paper to appear soon. Nonetheless, we can, very generally, state that the conclusions that we arrive at are consistent with our expectations in most cases. For instance, self-adaptation yields "smoother" behavior; proportional selection is less good than ranked $(\mu + \lambda)$ schemes; uniform crossover favors convergence; hillclimbers are worse than GAs in general, etc. Moreover, we are not only able to validate our intuition, but to do so quantitatively. For example, not only do we know that VGA is better than CGA with *G(4)* generations, but *how much* better.

We are aware that statistical analysis of this kind do not highlight the particular cases which are often most interesting. However, they do allow us to affirm what we can expect from the algorithms *in general* with a high degree of confidence and, furthermore, establish bounds on the worse case expected performance of the said algorithms. If, as is our intent, we must select from a variety of possible algorithms in order to achieve efficiency, the proposed methodology yields reliable unbiased elements aimed at making the best choice.

Finally, we would like to comment as to how the results above relate to the No-Free-Lunch-Theorem (NFLT) [12] which, intuitively put, asserts that we should not expect any algorithm to be better than any other, in general. We are confronted here

with an apparent contradiction, for we claim that the results above apply, in general. Clearly, the values for κ and κ in any of the algorithms allow for their categorization. But the apparent contradiction is easily dispensed with because our functions do not have the scope required by the NFLT. Although we have worked with a very large functional space, no matter how large, it does not encompass *all* the possible functions. For instance, multivariate and constrained functions are not included, to mention only two important kinds which have been left out. Therefore, what interest does it have, if any, to apply the proposed methodology? In the past, most of the experimental analysis have been biased, demonstrative and of limited scope, so that there is no way to determine whether some of the conclusions may have been incorrect. Within the realm of Ξ we are able to avoid biased and merely qualitative conclusions. In that sense the proposed method is an improvement over others generally employed.

7 References

1. Mitchell, M., "What Makes a Problem Hard for a Genetic Algorithm? Some Anomalous Results and Their Explanation", Machine Learning, 13, 285-319, 1993.
2. Back, T., "The Interaction of Mutation Rate, Selection and Self-Adaptation Within a Genetic Algorithm", R. Maumnler and B. Manderick, editors: Parallel Problem Solving from Nature, 2, 85-94, Elsevier, Amsterdam, 1992.
3. Fogel, D., Ghozeil, A., "Schema Processing under Proportional Selection in the Presence of Random Effects", IEEE Transactions on Evolutionary Computation, Vol. 1:4, 290-293, 1997.
4. Kuri, A., and Galaviz, J., "Towards a New Framework for the Analysis of Genetic Algorithms", International Computation Symposium, I.P.N., México, 1998.
5. De Jong, K., Sarma, J., "An Analysis of the Effects of Neighborhood Size and Shape on Local Selection Algorithms," Proc. of PPSN-96, the Second International Conference on Parallel Problem Solving from Nature, Springer-Verlag, 1996.
6. Mitchell, M., Forrest, S., and Holland, J., "The Royal Road for Genetic Algorithms: Fitness Landscapes and GA Performance", in F.J. Varela and P. Bourgine, eds., *Towards a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*, MIT Press, 1992.
7. Schmitzberger, P., "Approximation and Interpolation of High Dimensional Functions by generalized Walsh Polynomials", in R. Trobec et al., eds., *Proceedings of the International Workshop Parallel Numerics '96*, Gozd Martuljek, Slovenia, 150-164, 1996.
8. Mitchell, M., *An Introduction to Genetic Algorithms*, 4:129, MIT Press, 1996.
9. Goldberg, D., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addision-Wesley, 1989.
10. Kuri, A., "A Statistical Genetic Algorithm", National Computation Meeting, Hidalgo, México, 1999.
11. Kuri, A., and Villegas, C., "A Universal Genetic Algorithm for Constrained Optimization", 6th European Congress on Intelligent Techniques and Soft Computing, Aachen, Germany, 1998.
12. Wolpert, D., and Macready, W., "No Free Lunch Theorems for Optimization", IEEE Transactions on Evolutionary Computation, 1:67-82, 1997.