Empirical Study:
Initial Population Diversity and Genetic Algorithm Performance

Pedro A. Diaz-Gomez and Dean F. Hougen
Robotics, Evolution, Adaptation, and Learning Laboratory (REAL Lab)
School of Computer Science, University of Oklahoma, OK, USA
pdiazg@ou.edu - hougen@ou.edu

Abstract

This article presents an empirical study regarding the hypothesis that higher diversity in initial populations for Genetic Algorithms can reduce the number of iterations required to reach an optimum and potentially increase solution quality. We develop the empirical study using some theoretical functions addressed by other researchers such that the input to the Genetic Algorithm is populations of differing diversity. It is expected that the effort in analyzing the initial population with a diversity measure is going to be compensated for by reducing the number of iterations required and perhaps improving solution quality.

1. Introduction

Genetic Algorithms (GAs) have been used for optimization, automatic programming, data analysis and prediction, genomics, evolutionary neural networks, and so forth [16]. Reducing computation time needed to reach optimal solutions would be beneficial. It is expected that if the initial population is more diverse, then the performance of the algorithm may be improved [3, 23].

Usually the initial population is generated randomly and sized empirically [6]. The use of diversity can help to address the population size, at least for problems where diversity can be determined [5]. Our purpose in this article is, then, to do an empirical study regarding the relationship between diversity in an initial population and GA performance. To do this, operators are going to be maintained as suggested by previous studies in order to see the impact of the input on the output.

2. Previous Work

It is recognized that diversity is important in evolutionary computation [3, 23, 11] both to avoid premature convergence [7, 13] and as a stopping criterion [2]. The literature regarding population size is rich [17, 18, 15, 14, 10, 12, 22, 7] and important because the initial population provides diversity to the GA [1, 18, 4]. If the population's diversity is not high enough, then it could be that an optimum cannot be reached [17, 18, 12, 11, 7]. Further, if the population is quite large then the algorithm could expend more computation time in finding solutions [15, 14, 10, 12, 11]. Additionally, the quality of the input is quite important. The initial population problem is to provide the building blocks necessary to solve the problem [8]; if there are not enough building blocks, then it is almost impossible for the algorithm to reach the goal [10].

Many efforts have been made toward solving the problem of population size but, because population size depends in part on the difficulty of the problem to be solved [17, 22, 10], it remains an open problem [18]. However, as population size is quite important for the efficiency of evolutionary algorithms [11, 18, 4], various empirical methods have been proposed and some success reported [6], like the use of self adaptation [10, 21], which basically uses varying population size and may be the most prominent result until now.

3. Diversity Metrics Used

Diversity measures the variety of or difference between objects compared. Diversity can be measured at the gene-level, chromosome-level, and population-level and various metrics may be used to measure diversity at these levels [5].

Gene-Level Diversity. *Entropy* is defined to be

$$H(p) = \frac{1}{l}\sum_{j=1}^{l} H_j \tag{1}$$

where p is the population, l is the chromosome length, and H_j is the Shanon Entropy [20] for locus j of the entire population. $H_j = -\sum_{i=1}^{N} p_{ij} * log_2 p_{ij}$ where N is the population size. The range of diversity values at the gene-level is $[0, 1]$, where 1 is greater diversity.

#	*Code/Chromosome*
1	1 0 1 1 1 0 0 0 1 0 0
2	0 1 0 1 1 1 0 0 0 1 0
3	0 0 1 0 1 1 1 0 0 0 1
4	1 0 0 1 0 1 1 1 0 0 0
5	0 1 0 0 1 0 1 1 1 0 0
6	0 0 1 0 0 1 0 1 1 1 0
7	0 0 0 1 0 0 1 0 1 1 1
8	1 0 0 0 1 0 0 1 0 1 1
9	1 1 0 0 0 1 0 0 1 0 1
10	1 1 1 0 0 0 1 0 0 1 0
11	0 1 1 1 0 0 0 1 0 0 1

Table 1. Case I: Structured Initial Population. Perfect Population Diversity, Center of Mass $(\overline{\mathbf{x}}_{\mathbf{1}}, \overline{\mathbf{y}}_{\mathbf{1}}) = (\overline{\mathbf{x}}_{\mathbf{0}}, \overline{\mathbf{y}}_{\mathbf{0}}) = (\mathbf{6}, \mathbf{6})$.

Chromosome-Level Diversity. The *neighborhood average* is defined to be

$$\mathcal{N}(C_m) = \frac{\sum_{k=0}^{l} |N_k(C_m)| * k}{\sum_{k=0}^{l} |N_k(C_m)|} \tag{2}$$

where C_m the *pivot* chromosome for which the measure is calculated, $N_k(C_m) = \{C_j \in \{0,1\}^l | \rho_H(C_m, C_j) = k\}$, ρ_H is the Hamming distance, and $|N_k(C_m)|$ the cardinality of $N_k(C_m)$, i.e., the number of neighbors of C_m at a Hamming distance k in the population [2].

The range of diversity values at the chromosome-level is $[0, l]$, where l is higher; however, theoretical work suggests that values greater than $l/2$ could be considered “good” for some problems [5].

Population-Level Diversity. The *center of mass* for x with respect to 1 is defined to be

$$\overline{x}_1 = \frac{\sum_{i=1}^{N} \sum_{j=1}^{l} j}{T} \tag{3}$$

where j is the column position where the allele has value 1, and T is the number of those alleles[1]. Equation 3 is, then, summing the coordinate positions j (row by row for each column with respect to an origin $(0,0)$, where alleles are $1's$ in the entire population) and averaging by the number of those points. Formulas for the y coordinate with respect to 1, and for the center of mass with respect to 0, are similar.

It seems that a good diversity value is reached when $(\overline{x}_1, \overline{y}_1) \approx (\overline{x}_0, \overline{y}_0) \approx (\frac{l}{2} + \frac{1}{2}, \frac{M}{2} + \frac{1}{2})$ [5]. However, just one dimension (i.e., $\overline{x}$ or $\overline{y}$) can be evaluated and, in each dimension, 1 or 0 or both may be considered.

[1]The initial population can be seen as a Cartesian system with the origin $(0,0)$ at the top left corner. The first gene is at position $(1,1)$.

#	*Code/Chromosome*
1	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2	0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
3	0 0 0 1 1 1 1 0 0 0 0 1 1 1 1
4	0 0 0 1 1 1 1 1 1 1 1 0 0 0 0
5	0 1 1 0 0 1 1 0 0 1 1 0 0 1 1
6	0 1 1 0 0 1 1 1 1 0 0 1 1 0 0
7	0 1 1 1 1 0 0 0 0 1 1 1 1 0 0
8	0 1 1 1 1 0 0 1 1 0 0 0 0 1 1
9	1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
10	1 0 1 0 1 0 1 1 0 1 0 1 0 1 0
11	1 0 1 1 0 1 0 0 1 0 1 1 0 1 0
12	1 0 1 1 0 1 0 1 0 1 0 0 1 0 1
13	1 1 0 0 1 1 0 0 1 1 0 0 1 1 0
14	1 1 0 0 1 1 0 1 0 0 1 1 0 0 1
15	1 1 0 1 0 0 1 0 1 1 0 1 0 0 1
16	1 1 0 1 0 0 1 1 0 0 1 0 1 1 0

Table 2. Case II: Structured Initial Population. Perfect Gene Diversity, $\mathbf{H}(\mathbf{P}(\mathbf{0})) = \mathbf{1.0}$.

4. Empirical Study

The input to a GA is an initial population, so it would be useful to test how the initial population affects the performance in terms of the number of iterations needed to reach an optimum and the quality of solutions found.

4.1. Error-Detecting Codes

Reeves [19] presents an interesting case of a structured initial population for error detecting. Here the population size is small and the initial random generation of the population is not sufficient for the solution of the problem. The populations in Tables 1 and 2 show two cases [19].

In Table 1 each column and each row has 5 ones and 6 zeros. Diversity measured at the gene-level is 0.9940. At the chromosome-level, the diversity is 6. Finally, at the population-level, $(\overline{x}_1, \overline{y}_1) = (\overline{x}_0, \overline{y}_0) = (6, 6)$. Looking at the gene-level, the measure is above 0.99; with a population size of 10, that can be considered good. This fact is reinforced with the values obtained at the chromosome-level and population-level being “perfect” (see Section 3). The pivot used in Equation 2 was the fifth member of the population in Table 1; it turns out that any pivot used has 10 neighbors at a distance of 6.

For comparison, 11 codes of length 11 were generated randomly. Table 3 shows the corresponding diversity values. None score as high in diversity as the ones obtained with the structured population. For example, $\overline{x}_0$ was always equal to 6.00 but that never was the case for $\overline{x}_1$[2].

[2]Not all $\overline{x}_1$ are in Table 3; however, none of them is equal to $\overline{x}_0$.

Metric	*Maximum*	*Minimum*	*Average*	*Std. Dev.*
Entropy	0.98	0.83	0.93	0.03
Neighbors	5.44	4.60	5.33	0.15
(x_1, y_1)	(6.61,6.83)	(5.43,5.39)	(6.07,5.96)	(0.28,0.35)
(x_0, y_0)	(6.00,6.62)	(6.00,5.38)	(6.00,6.06)	(0.00,0.32)

Table 3. Measuring a Random Initial Population of 11 Individuals. Chromosome Length 11. 30 Runs.

Metric	*Maximum*	*Minimum*	*Average*	*Std. Dev.*
Entropy	0.99	0.91	0.95	0.02
Neighbors	6.35	5.73	6.07	0.20
(x_1, y_1)	(8.47,9.38)	(7.38,8.10)	(7.99,8.59)	(0.31,0.31)
(x_0, y_0)	(8.00,8.92)	(8.00,7.67)	(8.00,8.42)	(0.00,0.30)

Table 4. Measuring a Random Initial Population of 16 Individuals. Chromosome Length 15. 30 Runs.

In Table 2, the population has 8 ones and 8 zeros in each column, and 8 ones and 7 zeros in each row except for row 1 which has all zeros. Diversity measures at the gene-level result in a perfect value of 1.00; at the chromosome-level the diversity value is 8 (it should be noted that all members of the population are at a Hamming distance of 8, independent of the pivot used); and, at the population-level $(\overline{x}_1, \overline{y}_1) = (8, 9)$ and $(\overline{x}_0, \overline{y}_0) = (8, 8)$.

For comparison, 16 individuals of length 15 were generated randomly. The corresponding diversity values are in Table 4. There were not "good" diversity values like the ones obtained with the structured population in Table 2. For instance, the maximum gene-level diversity was 0.99 vs. 1.00 for the structured.

4.2. The One-Max Function

A common theoretical problem for GAs is to find a chromosome of all ones starting from an initial random population. Harik and Lobo [10] tested the one-max function using an optimal GA in order to compare results with a "parameter-less genetic algorithm." The optimal GA has the following parameters with no mutation involved: chromosome length 100, tournament size 2, and probability of crossover 1 [10]. These authors reported, on average, $2,500$ function evaluations with the optimal GA and $7,400$ with the "parameter-less genetic algorithm," performing 20 runs.

Our test is performed over 30 runs, i.e., using 30 different seeds. The tests run the algorithm until it reaches the maximum or gets stuck, i.e., a column of zeros is encountered. Figure 1 shows how the entropy measure influences

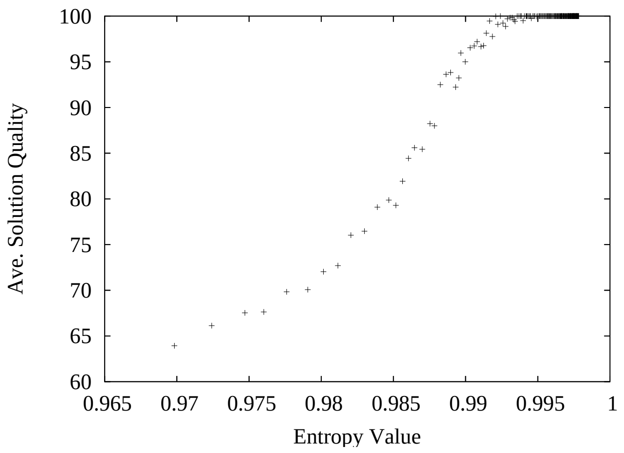

Figure 1. Entropy Measure vs. Quality of the Solution. One-Max Function. 30 Runs.

the average quality of the solution. For entropy values less than 0.98299 (population size less than 42) almost no run finds the optimum. For entropy values between 0.983887 and 0.993427 (population sizes between 44 and 110), at least one run finds the optimum. For entropy values between 0.993573 and 0.995 (population sizes between 112 to 144), almost all find the optimum. Finally, the algorithm always converges to the maximum for entropy values averaging above 0.995 for the test set described previously.

4.3. Deceptive Functions

Deceptive functions have at least one heavy attractor towards a local minimum [9], such as

$$\begin{array}{ll} F(I) = 4 * (\#Ones - 75), & \text{if } \#Ones >= 75 \\ F(I) = -\#Ones + 75, & \text{otherwise.} \end{array} \tag{4}$$

Similar tests to those in Section 4.2 were performed with a chromosome size of 100, 30 runs, (i.e., using seeds from 1 to 30), selection pressure of 2, crossover probability of 1.0, a mutation rate per bit of 0.002, and two stop criteria: when the algorithm reaches a local (or global) maximum or when gene-level diversity reaches 0. Figure 2 shows the corresponding relationship between entropy and average solution quality. The algorithm converges to the local maximum.

As genes were generated uniformly randomly, almost all the individuals are in the neighborhood of 50, and effectively the metric, as in Equation 2, measures an average of $\overline{\mathcal{N}}(C_m) = 49.99$ with a standard deviation of 0.019, over population sizes from 24 to 100. If that is the case, then it is expected that almost all the members of the population are going to lie in the neighborhood of the local maximum according to Equation 4. The algorithm is then trapped in the local maximum, as the empirical study shows.

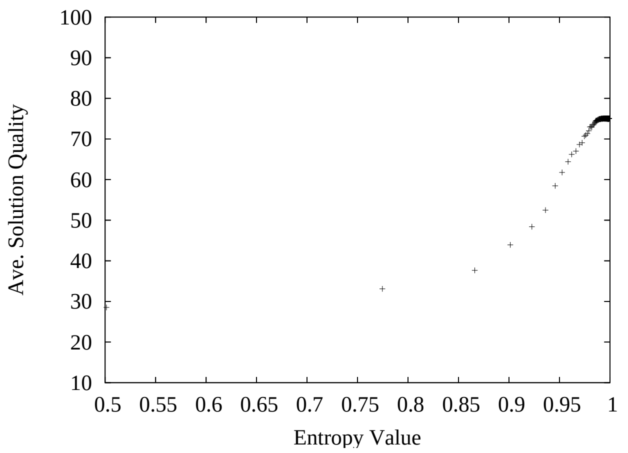

Figure 2. Entropy Measure vs. Average Quality of the Solution. Population Size $\geq 2 \leq 722$. Equation 4. Average on 30 Runs.

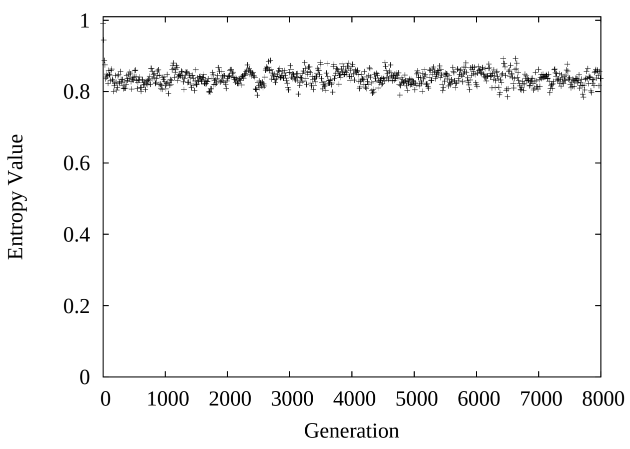

Figure 3. Entropy Variation when Fitness Set to 75 when $\#\mathrm{Ones} < 75$. Equation 5. Step Size 10 Generations.

In order to analyze the initial random population for this case, it is interesting to know the number of ones generated for each chromosome and take the maximum. For population size 100 the max(max) over 30 runs was 68, with a standard deviation of 1.88. As the max(max) obtained is less than 75 then there is no chance for the GA to climb to the global maximum according to Equation 4 with the actual set of parameters. However, one possible question that can arise now is, can the number of individuals in the initial population and selection pressure help Equation 4 to climb to the global optimum with the same operators? Tests were performed over 100, $1,000$ and $5,000$ individuals with the same 30 seeds and a tournament selection pressure of 70. There was no improvement in the quality of the solution, nor in the average of the number of function evaluations.

Two more deceptive functions were tested in order to check the impact of selection pressure in finding the global maximum. Equation 4 was changed in the range of $\#Ones < 75$ as in Equations 5 and 6.

$$\begin{array}{l} F(I) = 4 * (\#Ones - 75), \ \text{ if } \#Ones >= 75 \\ F(I) = 75, \ \text{ otherwise} \end{array} \tag{5}$$

$$\begin{array}{l} F(I) = 4 * (\#Ones - 75), \ \text{ if } \#Ones >= 75 \\ F(I) = \#Ones, \ \text{ otherwise} \end{array} \tag{6}$$

First, the fitness function is set as a constant equal to 75 for individuals with $\#Ones < 75$ (see Equation 5); second, the fitness function is set to $\#Ones$ in the same range (see Equation 6). With 100 uniform randomly generated chromosomes, with a selection pressure of 100, probability of crossover of 1 and mutation of 0.01, the algorithms diverged. The selection pressure was chosen as 100 just to show that even with the highest selection pressure, the deceptive functions presented prevent the GA from climbing to the global maximum. Tests were repeated with a mutation of 0.002 and the behavior of the algorithm was the same. Figure 3 shows the variability of entropy when the fitness function is set to 75 for $\#Ones < 75$. The same variability of entropy was observed using Equation 6. The algorithm diverged in both cases for the test set established.

5. Analysis of Results

For the error detection codes presented in Section 4.1, the population in Table 1 shows good diversity measures at the chromosome-level and the population-level. The question, then, is why was the gene-level diversity measure not perfect (i.e., equal to 1)? It is because each column has 5 ones and 6 zeros—the population size is 11. In Table 2 (population size 16 and code length 15) a perfect score is obtained at the gene-level (each column has 8 ones and 8 zeros), a perfect score is obtained at the chromosome-level (average Hamming distance is equal to $8 = l/2 + 1/2$), and a perfect diversity measure is obtained at the population level with respect to 0. However, with respect to 1 the result is $(\overline{x}_1, \overline{y}_1) = (8, 9)$, i.e., the metric shows that there are more ones located toward the "bottom" of the population (y coordinate). This is true because the first row in Table 2 is all zeros and the last one is all ones (as it was built [19]).

For the one-max function presented in Section 4.2, as population size grows and, likewise, diversity values increase, the quality of the solution is improved. However, if

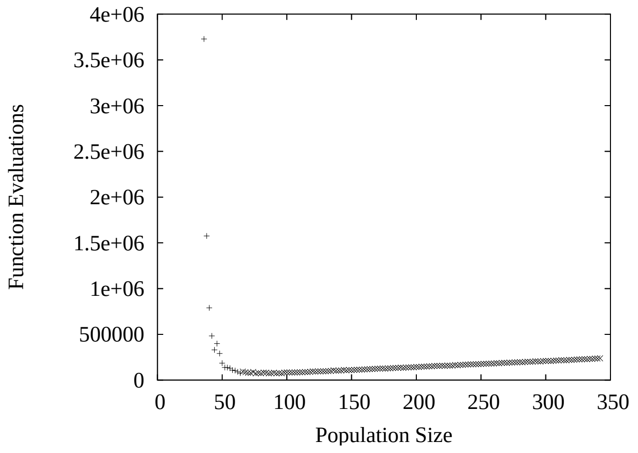

Figure 4. Population Size vs. Function Evaluations. Population Size ≥ 30 and ≤ 342. 30 Runs.

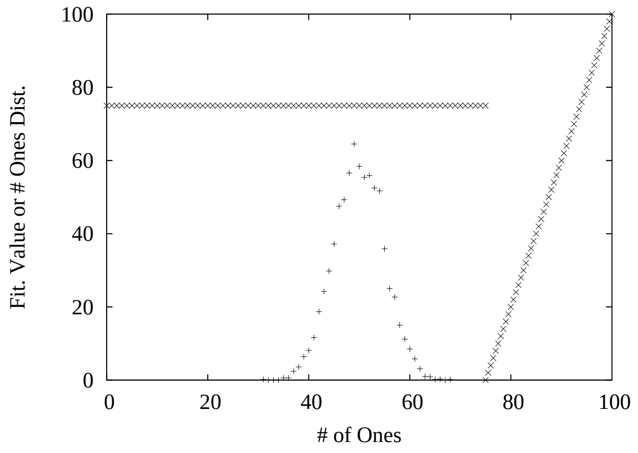

Figure 5. Equation 4 with Fitness Set to 75 when $\#\mathrm{Ones} < 75$. Distribution of Number of Ones for Population Size 100 on 30 runs.

the population size is greater than 100, and likewise the entropy value, then the number of function evaluations grows linearly, as is shown in Figure 4, where a population size of 342 was selected[3]. So, in this case, a "good" entropy value (0.997) was not good for the performance of the algorithm in terms of the number of function evaluations to reach the global optimum. Having more individuals in the population, above 100, just gives more function evaluations to the algorithm. However, in those cases, there is almost a guarantee of convergence to an optimum, as was stated in Section 4.2.

Besides our interest in knowing the general trend for population sizes greater than 100, it would be nice to know what happens to the linear trend that Figure 4 has when the population size is less than 100. To do this, the GA continues with the same parameters, the same 30 runs, and when the algorithm reaches an entropy of 0 with a specific seed, then the population is reinitialized using the same seed and the same number of individuals. It should be emphasized that in the reinitialization the pseudo-random generator is not reinitialized because if it was reinitialized then the algorithm is going to reach the same state again. The pseudo-random generator creates new populations until the GA arrives at an entropy of 0 or reaches the optimum. The number of function evaluations is counted until the algorithm reaches all ones. For population sizes less than 64 there is an incremental increase in the number of function evaluations (see Figure 4), entropy values range, on average, under 0.988. For population sizes less than 30 (entropy values less than 0.976), it is possible that the algorithm diverges for some seeds. For example, tests with seed 1 and population sizes 20 and 22 have taken the algorithm more than 4 trillion function evaluations without converging.

For the deceptive function used in Equation 4, note the same tendency in the number of function evaluations as with the one-max function (see Figure 7). For population sizes less than 64 (entropy value 0.988) it is possible that the algorithm expends a lot of function evaluations, and for population sizes less than 22 (entropy value 0.96634) it is possible that the algorithm diverges with the parameters previously defined. For population size 20, for example, the algorithm performed more than 44 million function evaluations without reaching an optimum.

As stated in Section 4.3, for deceptive functions it is unlikely the GA will climb to the global optimum, unless there are some individuals to compete with the ones in the neighborhood of 50. If, for example, we think of the probability of having an individual with 75 ones in a 100 population, 0.0004974% is obtained as an upper bound, i.e., 0.004974% in a population of $1,000$, 0.02487% in a population of $5,000$, 0.4974% in a population of $100,000$ and so forth. In order to confirm the last statement, one finds the maximum number of ones generated in the first generation for a population of size $1,000,000$ taking into account 30 seeds. The max(max) was 75. (See Figure 6 for the distribution of max(max) in the first generation). The first chromosome with length 75 appears in a population of $600,000$ and it is in great disadvantage compared with all the chromosomes because their fitness value is just 0.

[3]At population size 342 the average entropy value is 0.997, a value that could be considered near to 1, which is the optimum.

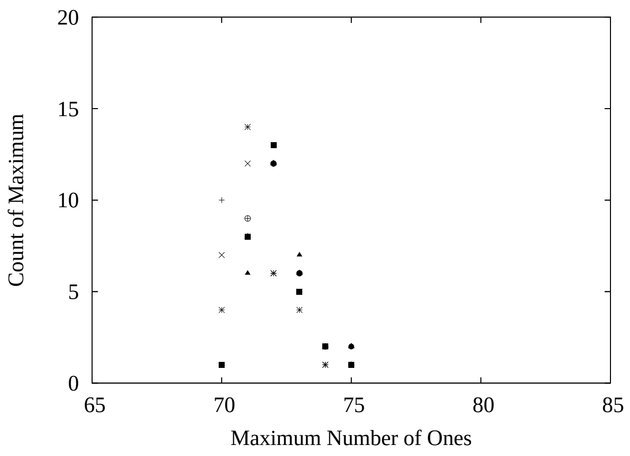

Figure 6. Distribution of Maximum Number of Ones for Population Sizes from $200,000$ to $1'000,000$. 30 Different Seeds.

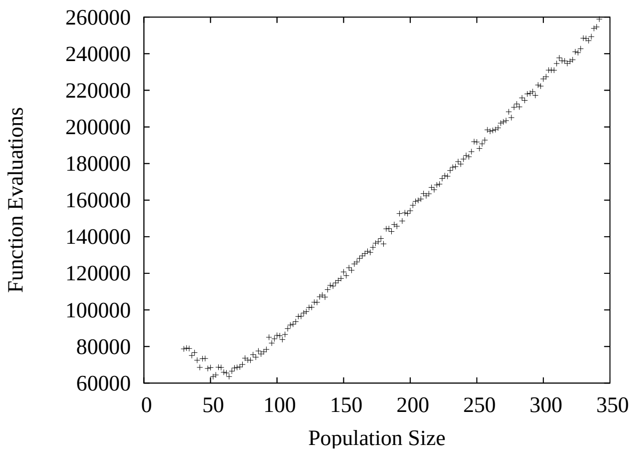

Figure 7. Population Size vs. Av. Function Evaluations. Population size ≥ 30 and ≤ 342. Equation 4. Average on 30 seeds.

6. Possible Solutions

Equation 4 and variants such as Equations 5 and 6 do not reach a global maximum in affordable computation time for the test set established in this empirical study. Equation 4 has a great attraction towards the local minimum. Equation 5, has a constant for $\#Ones \leq 75$, and consequently and no way to determine the "good" individuals in that region. Figure 5, besides showing Equation 5, shows the concentration of chromosomes around the center for a uniform randomly distributed initial population. Equations 5 and 6 have a discontinuity at 75, making some individuals in the range of $\#Ones > 75$ at a disadvantage in fitness value compared with the ones at $\#Ones \leq 75$.

Another factor that directly influences the convergence to the global optimum is the initial distribution of individuals around the center (see Figure 5). This distribution in the initial population marks a locality in the search space that is difficult to resolve with GA operators for these type of fitness landscapes. If two individuals whose genes have been generated uniformly randomly (around the center) are selected to cross over, it is expected that their offspring are going to be around the center too. Selection chooses the best, and this is true for the case of Equations 4 and 6, but in those cases the gravity is around the local minimum; for the case of Equation 5 there is almost no attraction at all.

To address these problems, changes can be proposed in almost all GA parameters and operators. However, as this article focuses on the initial population, our approach here looks principally at the distribution of the initial population.

6.1. Uniform Distribution

As stated before, the initial random generation of $\#Ones$ is uniform, i.e., each chromosome has on average 50 ones and 50 zeroes. There is, then, no uniform distribution of individuals all across the fitness landscape, i.e., between 0 and 100, so that some "good" individuals can climb to the global maximum.

For this test, 100 individuals were generated taking into account the fitness landscape as shown in Figure 8. A "+" corresponds to individuals uniformly generated according to the fitness landscape and "$\times$" corresponds to genes uniformly generated in the population. Tests were performed 30 times with 100 individuals each time. 19 zeros and 17 hundreds were obtained in the $3,000$ initial individuals generated. The entropy measure increases for all population sizes above 40 when a uniform fitness landscape distribution is used compared with a random gene distribution.

For Equation 4, the GA obtained the global maximum 13 times, the local maximum 12 times, and values between 92 and 96 5 times. On average it obtained a 88.93 solution quality (with $stdv = 11.81$) using on average 742.10 function evaluations. The quality of the solution improved as shown in Figure 9 compared with Figure 2. Equations 5 and 6 may not converge with the current set of parameters. For Equation 5, the 75% that dominates, with a fitness value of 75 dominates 18.75% more, so just 6.25% of the seach space can compete with the rest. For Equation 6, the same thing happens. For Equation 5 the selection pressure was changed from $S_p = 2$ to $S_p = 8$ (the selection pressure was increased until all runs converged or the diversity was equal to zero). In this case, 25 global maximums were obtained,

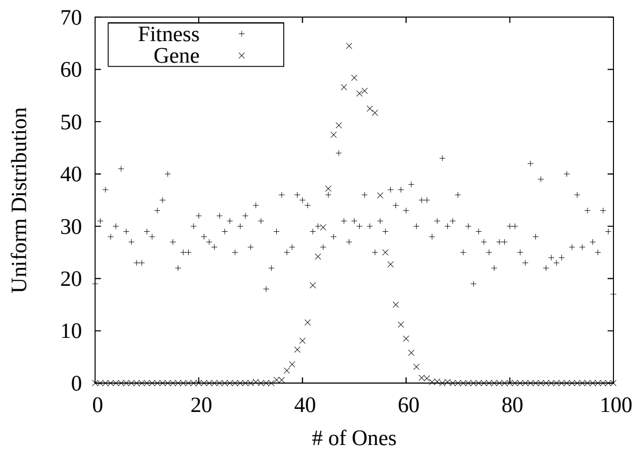

Figure 8. Population Uniformly Distributed between Fitness Values 0 and 100 and Uniformly Distributed at the Gene Level. 30 Runs.

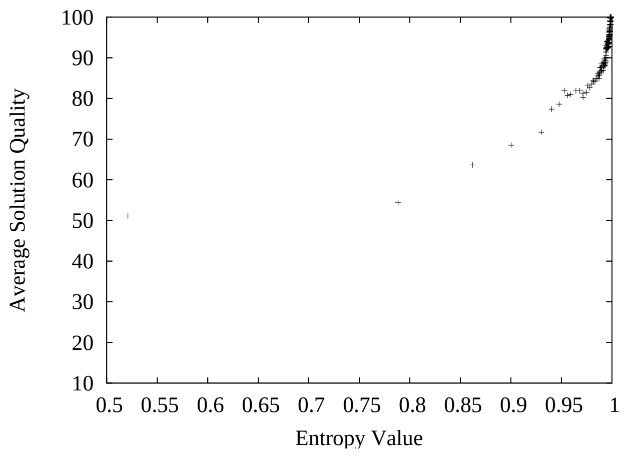

Figure 9. Entropy Measure vs. Average Quality of the Solution. Population Size $\geq 2 \leq 722$. Equation 4. Uniform Fitness Landscape Distribution. Average on 30 Runs.

no local maximum was obtained, and 5 times it reached values between 92 and 96. On average, the quality of the solution was 98.93 (with $stdv = 2.55$) and 235.80 function evaluations. For Equation 6, a selection pressure of 7 was needed for the GA to reach the global maximum in all runs; it spent on average 246.00 function evaluations.

6.2. Artificial Insemination

For deceptive functions like Equation 4, or with a great valley as in Equation 5, or with discontinuities like Equations 5 and 6, there may not be fit individuals when their genes are uniformly random. Therefore, we propose the use of *artificial insemination*, which means the insertion of one or more fit individuals into the current population—when one or some of those are known.

We use an artificial population of 5 individuals with 94 alleles of type 1 (just at the point where individuals can compete) and one is randomly selected and crossed over with a probability of 25%. So, 75% of crossovers are expected to be between two individuals normally selected, and 25% between one belonging to the artificial population and one selected normally.

For Equation 4, the GA found the global maximum 13 times. It had an average solution quality 85.83 (with $stdv = 12.6$) and spent $173,682.96$ function evaluations (with $stdv = 217,073.91$). For Equation 5, the GA always found the global maximum; it had an average of $5,334.33$ function evaluations (with $stdv = 1,393.65$). For Equation 6, the GA always found the global maximum; it had on average $2,385.86$ function evaluations (with $stdv = 257.75$).

7. Conclusions and Future Work

In this article some case studies were presented regarding the influence of diversity in the performance of GAs. The case of the error-coding examples [19], could be seen in two ways. As a counter example to random initial populations, in the sense that the error-coding population was structurally generated, and as a perfect example of diversity because the error-coding has "optimum" diversity values according to the metrics presented in Section 3. It seems that for small populations it may be better to generate structured chromosomes than random ones [19], and diversity can help to measure how structured the initial population is.

For the cases of the one-max function and deceptive function as in equation 4, it is highlighted that a minimum initial population is needed in order to converge—in our experimental set up we obtain 30 individuals, i.e, an average entropy of 0.97—however, there is approximately an optimum size in the neighborhood of 100 for the one-max function [10] and 64 for the deceptive one, in the sense that, usually, fewer individuals or more individuals than that neighborhood can cost more function evaluations to reach an optimum (global for the case of the one-max and local for the case of the deceptive function) or the divergence of the algorithm (see Figures 4 and 7).

Deceptive Equation 4 and its variants presented in Section 4.3 show a drawback in the random generation of the initial population, in the sense that if almost all chromosomes are going to be in the same neighborhood, i.e., where a local maximum is located, the GA is going to be trapped in it (see Figure 5). This fact enforces the hypothesis that if there is no "good" diversity in the search space, it is possible for the algorithm to diverge or find a poor solution. However, this does not imply, in general, that a higher diversity automatically gives us a better performance in GAs. This is a natural consequence of the "optimal neighborhood" of population size where fewer or more individuals than the "optimal" can cause the algorithm to diverge, expend more computations, or find poor solutions. Figure 6 shows us the paradox: it is possible that not even with $1,000,000$ individuals in the population will the GA reach the global optimum because at that population size the maximum number of ones obtained in the chromosomes are less than or equal to 75, exactly where the inflexion or discontinuity occurs. Neither the crossover and mutation operators, nor the selection pressure, could produce the necessary changes to reach the global optimum when the population was created using random generation of genes. In order to get out of the trap, the initial random generation had to be changed taking into account a uniform distribution of fitness values in conjunction with a higher selection pressure.

One important topic not covered in the present study, is the computational complexity spent in the calculation of diversity. For the metrics presented in Section 3, the computational complexity is $\mathcal{O}(Nl)$. This is the cost of the analysis when the metric approach suggested is used.

This article was focused on the importance of the input for GAs; however, it should be taken into account that the setting of the rest of the GAs' parameters influence the performance of the algorithm as well.

References

[1] J. T. Alander. On optimal population size of genetic algorithms. In *IEEE Computer Systems and Software Engineering*, pages 65–69, 1992.

[2] T. Bäck. *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, 1996.

[3] E. K. Burke, S. Gustafson, and G. Kendall. Diversity in genetic programming: An analysis of measures and correlation with fitness. *IEEE Transactions on Evolutionary Computation*, 8(1):47–62, 2004.

[4] J. C. Costa, R. Tavares, and A. Rosa. An experimental study on dynamic random variation of population size. In *IEEE International Conference on Systems, Man, and Cybernetics*, pages 607–612, 1999.

[5] P. A. Diaz-Gomez and D. F. Hougen. Initial population for genetic algorithms: A metric approach. Submitted to *International Conference on Genetic and Evolutionary Methods*, 2007.

[6] A. E. Eiben, R. Hinterding, and Z. Michalewicz. Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 3(2):124–141, 1999.

[7] W. G. Frederick, R. L. Sedlmeyer, and C. M. White. The hamming metric in genetic algorithms and its application to two network problems. In *Proceedings of the ACM*, pages 126–129, 1993.

[8] D. E. Goldberg, K. Deb, and J. H. Clark. Genetic algorithms, noise, and the sizing of populations. *Complex Systems*, 6:333–362, 1992.

[9] D. E. Goldberg, K. Deb, and J. Horn. Massive multimodality, deception, and genetic algorithms. Technical Report 92005, Illinois Genetic Algorithms Laboratory, University of Illinois, 1992.

[10] G. R. Harik and F. G. Lobo. A parameter-less genetic algorithm. In *Genetic and Evolutionary Computation Conference*, pages 258–265, 1999.

[11] Z. M. Jaroslaw Arabas and J. Mulawka. GAVaPS—a genetic algorithm with varying population size. In *IEEE International Conference on Evolutionary Computation*, volume 1, pages 73–78, 1995.

[12] V. K. Koumousis and C. P. Katsaras. A saw-tooth genetic algorithm combining the effects of variable population size and reinitialization to enhance performance. *IEEE Transactions on Evolutionary Computation*, 10(1):19–28, 2006.

[13] Y. Leung, Y. Gao, and Z. B. Xu. Degree of population diversity—a perspective on premature convergence in genetic algorithms and its Markov chain analysis. *IEEE Transactions on Neural Networks*, 8(5):1165–1176, 1997.

[14] F. G. Lobo and D. E. Goldberg. The parameter-less genetic algorithm in practice. *Information Sciences—Informatics and Computer Science*, 167(1-4):217–232, 2004.

[15] F. G. Lobo and C. F. Lima. A review of adaptive population sizing schemes in genetic algorithms. In *Genetic and Evolutionary Computation Conference*, pages 228–234, 2005.

[16] M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, 1998.

[17] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz. Bayesian optimization algorithm, population sizing, and time to convergence. Technical Report 2000001, Illinois Genetic Algorithms Laboratory, University of Illinois, 2000.

[18] A. Piszcz and T. Soule. Genetic programming: Optimal population sizes for varying complexity problems. In *Generic and Evolutionary Computation Conference*, pages 953–954, 2006.

[19] C. R. Reeves. Using genetic algorithms with small populations. In *International Conference on Genetic Algorithms*, pages 92–99, 1993.

[20] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–656, 1948.

[21] B. Thomas, A. Eiben, and V. der Vaart. An empirical study on GAs without parameters. In *Parallel Problem Solving from Nature V*, pages 315–324, 2000.

[22] T.-L. Yu, K. Sastry, D. E. Goldberg, and M. Pelikan. Population sizing for entropy-based model building in genetic algorithms. Technical Report 2006020, Illinois Genetic Algorithms Laboratory, University of Illinois, 2006.

[23] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8(2):173–195, 2000.