ENTROPY AND GENETIC ALGORITHMS:
DEFINITION, AND SOME GRAPHS

Aristides T. Hatjimihail

Hellenic Complex Systems Laboratory
P.O. Box 56, GR-661 00
Drama, Greece.
e-mail: aris@isosun.ariadne-t.gr

Technical Report I
October 1993
A. Introduction
Although the entropy concept has been used extensively in the study of complex systems (1), it has received little attention in the theory of genetic algorithms (2).
The purpose of this note is to propose a definition and present some graphs of the entropy of the population of the strings, during the genetic search.

B. Definition of Entropy
On the analogy of the physical systems, we divide the genotypically defined phase space of the population of the strings into compartments, where the different points of a compartment represent phenotypically identical populations. These populations may differ genotypically.
The entropy of the population of the strings is a measure of the volume of the compartment containing the phase space point which represents the population (1).
The entropy \( E_i \) of a string is defined as:
\[
E_i = -p_i \cdot \ln p_i,
\]
where \( p_i \) is the phenotypic probability of the string.
Consequently the entropy of a population of \( n \) strings is defined as:
\[
E = \sum_{i=1}^{n} -p_i \cdot \ln p_i
\]

C. Objective Function
The objective function to be optimized was the 120-bit, multimodal, deceptive function, formed by summing 40 copies of the 3-bit deceptive function defined by Goldberg (3).

D. Initial Population
The initial population was set to 600.

E. Operators
The following operators were used:
- Crossover.
One-point crossover was used.

- Mutation.

The probability for mutation ($p_m$) is defined as probability per bit.

- Transposition.

Let us denote with $A_i$ the $i$th symbol of a string $A$, of length $l$. Assume that a substring of length $d$ of the string $A$ is transposed from the site $g$ to the site $h$. Then the string $A'$ will be generated. Assuming that $g > h$ and that $d < g - h$, we have:

\[
\forall j \in N : (1 \leq j < h) \Rightarrow A'_j = A_j \\
\forall j \in N : (h \leq j < h + d) \Rightarrow A'_j = A_x \text{ where } x = j + g - h \\
\forall j \in N : (h + d \leq j < g + d) \Rightarrow A'_j = A_y \text{ where } y = j - d \\
\forall j \in N : (g + d \leq j \leq l) \Rightarrow A'_j = A_j
\]

In a similar way, we can define transposition when $d > g - h$, when $g < h$, and so on.

The $g, h$, and $d$ are defined randomly.

The probability for transposition ($p_t$) is defined as probability per string.

F. Selection Schemes

The following selection schemes were used:

- Binary tournament selection (4).

The probability for selecting the fittest string was set to 1.0.

- Deterministic crowding (5).
- Sharing (6).

Sharing was based on the Hamming distance. $\sigma_{\text{share}}$ was set to $l/2 = 60$ and $\alpha$ to 1. The niche count was estimated by sampling the $1/10$ of the population.

G. Phenotypic Probability

The phenotypic probability $p_i$ of each string was calculated using a recursive algorithm.
H. Number of Generations
The genetic search continued for 250 generations.

I. The Graphs
The graphs of the entropy vs the genetic search for different combinations of selection schemes, probability for mutation, and probability for transposition, are presented.
The graphs of each front page present the maximum and mean fitness and the entropy of the population of the strings during five consecutive runs.
The graphs of each back page present the maximum and mean fitness and the entropy of the population of the strings during one randomly selected run.

J. Discussion
The graphs of the entropy vs the generation number show distinct patterns for each combination of selection scheme, probability for mutation, and probability for transposition. It can be easily seen that the genetic algorithms can create a great amount of order. Considering the optimization of our objective function, the entropy of the population of the strings, at the end of the genetic search, is about 30 orders of magnitude less than the entropy of the initial population. On the other hand, the graphs describe quantitatively the randomness that mutation and transposition reintroduce into the population.
Obviously, the graphs of the entropy vs generation give us more information about the process of the genetic search, than the commonly used graphs of the mean and maximum fitness. Therefore, the mathematical analysis of the entropy during the genetic search could offer a new insight into the nonlinear dynamics of the genetic algorithms and their operators. The analysis will be probably based on the schema theorem.
A particularly intriguing prospect is that such analysis could possibly show that the most efficient genetic search is performed at the boundary between order and randomness, as it happens to other complex adaptive systems (7).
I hope that this note will stimulate a creative exchange of ideas, in these directions.
H. Acknowledgments
I thank Theophanes T. Hatjimihail for the helpful discussions, during the preparation of this note.

I. References
Parameters of the genetic search:
\( p_c = 1.0, \ p_m = 0.0000, \ p_t = 0.000 \)

Selection scheme:
Binary tournament selection

Front page:

The results of five consecutive runs.
The optimal solution was found during 2/5 runs.

Back page:

The results of one run.
Fittest solution found: 1196, during the 118th generation.
Maximum and Mean Fitness (vs generation)

Entropy (vs generation)
Parameters of the genetic search:
$p_C = 1.0, \ p_m = 0.0000, \ p_t = 0.000$

Selection scheme:
Deterministic Crowding.

Front page:
The results of five consecutive runs.
The optimal solution was found during 0/5 runs.

Back page:
The results of one run.
Fittest solution found: 1194, during the 68th generation.
Maximum and Mean Fitness (vs generation)

ENTROPY (vs generation)
Parameters of the genetic search:
P_c = 1.0, \ p_m = 0.0000, \ p_I = 0.000

Selection scheme:
Sharing.

Front page:
The results of five consecutive runs.
The optimal solution was found during 0/5 runs.

Back page:
The results of one run.
Fittest solution found: 1168, during the 245th generation.
Maximum and Mean Fitness (vs generation)

ENTROPY (vs generation)
Parameters of the genetic search:
$p_c = 1.0, \ p_m = 0.0001, \ p_t = 0.000$

Selection scheme:
Binary tournament selection

Front page:

The results of five consecutive runs.
The optimal solution was found during 1/5 runs.

Back page:

The results of one run.
Fittest solution found: 1196, during the 107th generation.
Maximum and Mean Fitness (vs generation)

ENTROPY (vs generation)
Parameters of the genetic search:
\( p_c = 1.0, \ p_m = 0.0001, \ p_t = 0.000 \)

Selection scheme:
Deterministic Crowding.

Front page:

The results of five consecutive runs.
The optimal solution was found during 0/5 runs.

Back page:

The results of one run.
Fittest solution found: 1182, during the 69th generation.
Maximum and Mean Fitness (vs generation).

ENTROPY (vs generation)
Maximum and Mean Fitness (vs generation).

ENTROPY (vs generation)
Parameters of the genetic search:
$p_c = 1.0, \ p_m = 0.0001, \ p_t = 0.000$

Selection scheme:
Sharing.

Front page:

The results of five consecutive runs.
The optimal solution was found during 0/5 runs.

Back page:

The results of one run.
Fittest solution found: 1162, during the 250th generation.
Parameters of the genetic search:
$P_C = 1.0, \ P_m = 0.0000, \ P_t = 0.001$

Selection scheme:
Binary tournament selection

Front page:
The results of five consecutive runs.
The optimal solution was found during 5/5 runs.

Back page:
The results of one run.
Fittest solution found: 1200, during the 100th generation.
Maximum and Mean Fitness (vs generation)

ENTROPY (vs generation)
Parameters of the genetic search:
\( p_c = 1.0, \; p_m = 0.0000, \; p_l = 0.001 \)

Selection scheme:
Deterministic Crowding.

Front page:

The results of five consecutive runs.
The optimal solution was found during 5/5 runs.

Back page:

The results of one run.
Fittest solution found: 1200, during the 75th generation.
Maximum and Mean Fitness (vs generation)

ENTROPY (vs generation)
Parameters of the genetic search:
$p_c = 1.0$, $p_m = 0.0000$, $p_l = 0.001$

Selection scheme:
Sharing.

Front page:

The results of five consecutive runs.
The optimal solution was found during 0/5 runs.

Back page:

The results of one run.
Fittest solution found: 1176, during the 250th generation.
Maximum and Mean Fitness (vs generation)

ENTROPY (vs generation)
Maximum and Mean Fitness (vs generation)

ENTROPY (vs generation)