Some New Results for Multiple-valued Genetic Algorithms

T. C. Wesselkamper
Hunter College and
The Graduate School and University Center
The City University of New York

Joshua Danowitz
Hunter College
The City University of New York

Abstract

This paper describes each of the operations involved in a genetic algorithm: reproduction, mutation, and selection, and discusses each in the language of classical multiple-valued logic. The differences among forms of reproduction that have been used by various researchers are examined and the relative importance of each of the operations in searching for highly fit members of a population is evaluated. The role of mutation in ensuring the completeness of the set of genetic operators is established. A recently proposed form of selection is shown to force convergence of the genetic algorithm, independently of reproduction and mutation. Finally, the theorems developed are applied to practical problems in the use of genetic algorithms.

I. Motivation

At the 1994 ISMVL Cabrasawan and Wesselkamper reported on the use of genetic algorithm techniques in searching for functions over E(3) which are complete with constants and have small radii [3]. In the oral presentation of that research, certain behaviors of the genetic search process which the researchers could not explain were reported, viz.,

1.) changes in the rate at which mutation occurred, from 0.0 through 0.1 had no effect whatever on the convergence of the mean fitness value of the population; and

2.) the convergence of the mean fitness value of the population was logarithmic.

In this paper we report some new results which cast light on the two phenomena reported above. To do this we first need to recast the usual, probabilistic description of a genetic algorithm into the language of multiple-valued logic.

II. Genetic Algorithms

A. The Naive View

A genetic algorithm is a search strategy that acts on a population P of strings over some alphabet. There is, over this space P of strings, called chromosomes; some kind of a fitness function F which is real valued: F: S -> R. If, for strings x and y, F(x) > F(y), then x is said to be more fit than y. A string x for which F(x) is small is said to be unfit. The strings are conceived of as strings of genes and the search algorithm has three steps.

1.) **Reproduction:** Pairs of strings in a sample drawn from the population ("parents") are compared; based on some reproduction rule new strings ("children") are produced. The original reproduction rule cut each of the parent strings at the same randomly chosen point and formed two new strings by joining the head of the first to the tail of the second and the head of the second to the tail of the first.

2.) **Mutation:** With some fixed probability a gene is randomly changed. Probabilities as low as 0.001 and as high as 1/L, where L is the length of the string, have been used.

3.) **Selection:** A new population of parents is selected. In some research the children in the i^{th} generation are the parents in the $(i+1)^{st}$ generation. In other work at each generation the new population of parents is chosen to be the most fit elements chosen from the set theoretic union of the parents and their children. In another variation, each pair of parents produces two children. Of the set formed by the two parents and two children the two individuals with the largest fitness values go on to become parents in the next generation. This is very Oedipal.

B. The Multiple-valued Logic View

Let E(k) = {0, 1, ..., k-1} be a finite set of objects. A **relation of degree h** on E(k) is any subset R of the Cartesian product $(E(k))^h$. An element of the Cartesian product is an h-tuple and is called a **member** of the relation R. We use angle brackets to denote a member of a relation, thus $\langle a \rangle = \langle a_1, a_2, ..., a_h \rangle$ in R. Since each relation R is a set, no member of R is a duplicate of any other member of R. A relation of degree h with n members is ordinarily presented as an n by h array, R = (r_{ij}). If a relation R is the whole Cartesian product, it is called the **universal relation.** If a relation consists only of the members $\langle x, x, ..., x \rangle$, for each x in R, then

Proceedings of the 25th International Symposium on Multiple-Valued Logic (ISMVL '95)
0730-3157/95 $10.00 © 1995 IEEE

it is called the trivial relation. Let $|R|$ denote the number of members in the relation R. If R is a relation of degree h over E(k), and of f is a two-place function f: $E(k)^2 \rightarrow E(k)$ then **f preserves the relation R** if for all <a>, <b> in R, $<f(a_1,b_1)\ f(a_2,b_2)\ ...\ f(a_h,b_h)>$ is in R.

A two-place function f(x,y) over E(k) is said to be a **essential function** if it is not a function of a single variable and if each element of E(k) occurs in the operation table for f. A two-place function f(x,y) over E(k) is said to be a **reproduction function** if, for each x R E(k), f(x,x) = x. In other words, the reproduction functions are just the functions which have each point of E(k) as a fixed point. In E(2) there are four reproduction functions. Two are functions of a single variable, the two projections, $p_1(x,y) = x$ and $p_2(x,y) = y$, the other two are the functions commonly called "and" and "or".

There are $729 = 3^6$ reproduction functions over E(3), each having an operation table with three fixed points. Two of these, the two projections $p_1(x,y) = x$ and $p_2(x,y) = y$ are functions of a single variable. Each of the other 727 is an essential function.

A **reproduction operator of degree h** is a sequence of h reproduction functions. We use square brackets to denote a reproduction operator, $[q] = [q_1, q_2, ..., q_h]$. If <a> and <b> are members of a relation of degree h over E(k) and [q] is a reproduction operator then [q] operates on <a> and <b> componentwise, viz.,

$$[q]<a><b> = <q_1(a_1,b_1)\ q_2(a_2,b_2), ..., q_h(a_h,b_h)>.$$

If R is a relation of degree h over E(k), and [f] is a reproduction operator then **[f] preserves the relation R** if for all <a>, <b> in R, [f]<a><b> is in R.

III. Reproduction

A. Four Variants of Reproduction

1.) Simple Crossover

In the language of genetic algorithms, a relation of degree h is a population of chromosomes, each with h genes, that is, an h-tuple. Several different definitions of crossover have been used. The simplest and earliest is described in [1, p. 12]. An integer j is chosen in the closed interval [1, h]. In each of the two h-tuples:

$<a> = <a_1\ a_2\ ...\ a_j\ \mathbf{a_{j+1}\ ...\ a_h}>$ and
$<b> = <b_1\ b_2\ ...\ b_j\ \mathbf{b_{j+1}\ ...\ b_h}>$

the components to the right of position j are interchanged to produce the pair of offspring:

$<a'> = <a_1\ a_2\ ...\ a_j\ \mathbf{b_{j+1}\ ...\ b_h}>$ and
$<b'> = <b_1\ b_2\ ...\ b_j\ \mathbf{a_{j+1}\ ...\ a_h}>$.

If k = 3, h = 9, and j = 5, we might have:

<a> = <0 1 2 0 1 **2 0 1 2**> and
<b> = <2 1 0 2 1 **0 2 1 0**>,

which produce

<a'> = <0 1 2 0 1 **0 2 1 0**> and
<b'> = <2 1 0 2 1 **2 0 1 2**>.

In this simplest form of crossover genetic operators form sequences of the projections p1 and p2. In the example, the two operators are:

$[p] = [p_1\ p_1\ p_1\ p_1\ p_1\ p_2\ p_2\ p_2\ p_2]$ and
$[q] = [p_2\ p_2\ p_2\ p_2\ p_2\ p_1\ p_1\ p_1\ p_1]$ each applied to <a> and <b>.

For h = 9 there are nine pairs of reproduction operators of this type, always applied together, one pair for each of the values of j in the closed interval [1, 9]. If j = h the operators are [p1 p1 ... p1] and [p2 p2 ... p2]. There are 2h such reproduction operators, the pair of identity operators produce offspring which are exact copies of the parents. These identity operators are included if the value of j is chosen in the interval [1, h]. In this paper we denote by G_1 the set of simple crossover operators.

2.) Dual Crossover

Other researchers [5, 6] have generalized the simple genetic algorithm by choosing two integers j and j' (j < j') in the interval [0, h]. In this version of crossover the subsequences of chromosomes between j+1 and j', inclusively, are interchanged.

$<a> = <a_1\ a_2\ ...\ a_j\ \mathbf{a_{j+1}\ ...\ a_{j'}}\ a_{j'+1}\ ...\ a_h>$ and
$<b> = <b_1\ b_2\ ...\ b_j\ \mathbf{b_{j+1}\ ...\ b_{j'}}\ b_{j'+1}\ ...\ b_h>$

produce the pair of offspring:

$<a> = <a_1\ a_2\ ...\ a_j\ \mathbf{b_{j+1}\ ...\ b_{j'}}\ a_{j'+1}\ ...\ a_h>$ and
$<b> = <b_1\ b_2\ ...\ b_j\ \mathbf{a_{j+1}\ ...\ a_{j'}}\ b_{j'+1}\ ...\ b_h>$

If k = 3, h = 9, j = 3 and j' = 6, we might have:

<a> = <0 1 2 **0 1 2** 0 1 2> and
<b> = <2 1 0 **2 1 0** 2 1 0>.

which produce:

<a'> = <0 1 2 **2 1 0** 0 1 2> and
<b'> = <2 1 0 **0 1 2** 2 1 0>.

This is equivalent to the application of the pair of reproduction operators:

$[p] = [p_1\ p_1\ p_1\ p_2\ p_2\ p_2\ p_1\ p_1\ p_1]$ and
$[q] = [p_2\ p_2\ p_2\ p_1\ p_1\ p_1\ p_2\ p_2\ p_2]$.

As in the previous case, the identity operations, producing offspring which are exact copies of the parents, are included in this set. In the case j' = h this form of crossover is the same as simple crossover. In the case j = 0 and j' = h, [q] and [r] are the two identity operators. We denote the set of dual crossover operators by G_2.

3.) Generalized Crossover

Generalized crossover is more general form of dual crossover in which each position in the reproduction operator can be either p_1 or p_2. For example,

$[q] = [p_1\ p_2\ p_1\ p_2\ p_2\ p_2\ p_1\ p_2\ p_1]$ or
$[r] = [p_2\ p_1\ p_2\ p_1\ p_1\ p_1\ p_2\ p_1\ p_2]$.

When a pair of reproduction operators, $[q] = [q_1, ..., q_h]$ and $[r] = [r_1, ..., r_h]$, have the property that for all i, q_i, r_i in {p1, p2} and $q_i \neq r_i$, then [q] and [r] are called **twins.** There are 2^h reproduction operators of this more

Proceedings of the 25th International Symposium on Multiple-Valued Logic (ISMVL '95)
0730-3157/95 $10.00 © 1995 IEEE

general crossover type, sometimes, but not necessarily applied in twin pairs. The difference is in the number of off-spring produced in any generation. If reproduction operators are applied in pairs then in any generation the size of the population of off-spring is the same as the size of the population of parents. If not applied in pairs, then the size of the population of off-spring is half the size of the population of parents. We denote the set of generalized crossover operators by G_3.

4.) Generalized Reproduction

The most general form of reproduction requires only that at each position in the reproduction operator there be a reproduction function. For example,

$$[q] = [q_1\ q_2\ q_3\ q_4\ q_5\ q_6\ q_7\ q_8\ q_9],$$

where each q_i is any reproduction function. In this case a child inherits from its parents any values that the parents have in common, but can inherit any value in E(k) at positions in which the parents do not share a common value. This possibility has not been considered by classical genetic algorithm researchers, since it cannot occur over E(2).

We can summarize these four possible definitions by borrowing from the notation of formal languages to define a reproduction operator of order h. If g is a reproduction function, let $[g^1] = [g]$ and let $[g^h] = [gg^{h-1}]$. Using this convention, define four sets of reproduction operators as follows:

$$G_1 = \{[p_1^j p_2^{h-j}] \mid 0 < j \le h\} + \{[p_2^j p_1^{h-j}] \mid 0 < j \le h\};$$

$$G_2 = \{p_1^j\, p_2^{j'}\, p_1^{j''} \mid j, j', j'' \ge 0, j+j'+j'' = h\} + \{p_2^j\, p_1^{j'}\, p_2^{j''} \mid j, j', j'' \ge 0, j+j'+j'' = h\};$$

$$G_3 = \{[p_{\varphi(1)}\, p_{\varphi(2)} \cdots p_{\varphi(h)}] \mid \varphi(i) = 1 \text{ or } 2\};$$

$$G_4 = \{[g_1\, g_2 \ldots g_h] \mid \text{for all x in E(k)}, g_i(x,x) = x\}.$$

B. Equivalence Theorem

We wish to investigate the situation in which certain specific sets of reproduction operators are applied, firstly to a pair of h-tuples and, secondly, to all of the pairs of h-tuples in a relation R. It is trivially true that G_4 contains G_3, G_3 contains G_2, and G_2 contains G_1.

__Theorem 1:__ G_1 generates G_2 and G_3.

Proof: To show that G_1 generates G_2, let [q] be an arbitrary element of G_2 and consider three cases. Firstly, if [q] is the identity operator then [q] is in G_1. Otherwise, by the definition of G_2 there are integers j and j', $1 \le j < j' \le h$, such that the components $q_1 = \ldots = q_j = p_1$, $q_{j+1} = \ldots = q_{j'} = p_2$, $q_{j'+1} = \ldots = p_1$. Secondly, if j' = h, the [q] is also in G_1. Finally, if j' < h, define three operators in G_1 as follows:

$[q_1] = [p_1^j p_2^{h-j}]$, its twin $[r_1] = [p_2^j\, p_1^{h-j}]$, and $[q_2] = [p_1^{j'}\, p_2^{h-j'}]$, and let $\langle a\rangle$ and $\langle b\rangle$ be arbitrary elements of a relation R. Applying $[q_1]$ and $[r_1]$ followed by $[q_2]$ gives:

$$[q_1]\langle a\rangle\langle b\rangle = \langle a_1 \ldots a_j\, b_{j+1} \ldots b_h\rangle;$$

$$[r_1]\langle a\rangle\langle b\rangle = \langle b_1 \ldots b_j\, a_{j+1} \ldots a_h\rangle; \text{ and}$$

$$[q_2][q_1]\langle a\rangle\langle b\rangle[r_1]\langle a\rangle\langle b\rangle =$$

$$[q2]\langle a_1 \ldots a_j\, b_{j+1} \ldots b_h\rangle\langle b_1 \ldots b_j\, a_{j+1} \ldots a_h\rangle =$$

$$[p_1^{j'}\, p_2^{h-j'}]\langle a_1 \ldots a_j\, b_{j+1} \ldots b_h\rangle\langle b_1 \ldots b_j\, a_{j+1} \ldots a_h\rangle$$

$$= \langle a_1 \ldots a_j\, b_{j+1} \ldots b_{j'}\, a_{j'+1} \ldots a_h\rangle = [q]\langle a\rangle\langle b\rangle.$$

To show that G_1 generates G_3, let [q] be an arbitrary element of G_3 and consider two cases. Firstly, if [q] is the identity operator then [q] is in G_1. Otherwise,

$$[q] = [p_{\varphi(1)}\, p_{\varphi(2)} \cdots p_{\varphi(h)}]$$

Define the follow sequence of twin operators, all in G1:

$[q_1] = [p_{\varphi(1)}\, p_{\varphi(2)} \cdots p_{\varphi(2)}]$, and its twin $[r_1]$;

$[q_2] = [p_{\varphi(1)}\, p_{\varphi(1)}\, p_{\varphi(3)} \cdots p_{\varphi(3)}]$, and its twin $[r_2]$;

...

$[q_{h-1}] = [p_{\varphi(1)}\, p_{\varphi(1)} \cdots p_{\varphi(1)}\, p_{\varphi(h)}]$.

Let $A_1 = [q_1]\langle a\rangle\langle b\rangle$ and $B_1 = [r_1]\langle a\rangle\langle b\rangle$, and for $2 < i < h-1$, let $A_i = [q_i]A_{i-1}B_{i-1}$ and $B_i = [r_i]A_{i-1}B_{i-1}$. Repeated substitution yields $[q_{h-1}]A_{h-2}B_{h-2} = [q]\langle a\rangle\langle b\rangle$ and since each A_i and each B_i is obtained by repeated applications of elements of G_1, G_1 generates G_3.

To understand the implication of the theorem, suppose k = 3, h = 4, and that the relation R_0 which forms the initial population consists of the four 4-tuples:

<0 1 2 0>,
<0 0 0 1>,
<1 2 0 2>,
<2 0 1 0>.

Consider first the effect of the operators from the set G_1, above. The three pairs of genetic operators in G_1 are:

$[g_1] = [p_1p_2p_2p_2]$	$[g_3] = [p_1p_1p_2p_2]$	$[g_5] = [p_1p_1p_1p_2]$
and	and	and
$[g_2] = [p_2p_1p_1p_1]$	$[g_4] = [p_2p_2p_1p_1]$	$[g_6] = [p_2p_2p_2p_1]$.

Applying each pair of operators to each of the six pairs of parents produces the set $R_1 = \{[g_i]\langle x\rangle\langle y\rangle \mid i = 1, \ldots 6, \langle x\rangle, \langle y\rangle \text{ in } R_0\}$ of thirty-six children:

<0 0 0 1>	<0 1 0 1>	<0 1 2 1>
<0 1 2 0>	<0 0 2 0>	<0 0 0 0>
<1 0 1 0>	<1 2 1 0>	<1 2 0 0>
<2 2 0 2>	<2 0 0 2>	<2 0 1 2>
<0 2 0 2>	<0 1 0 2>	<0 1 2 2>
<0 0 1 0>	<0 0 1 0>	<0 0 0 0>
<1 1 2 0>	<1 2 2 0>	<1 2 0 0>
<2 0 0 1>	<2 0 0 1>	<2 0 1 1>
<0 0 1 0>	<0 1 1 0>	<0 1 2 0>
<0 2 0 2>	<0 0 0 2>	<0 0 0 2>
<1 0 0 1>	<1 2 0 1>	<1 2 0 1>
<2 1 2 0>	<2 0 2 0>	<2 0 1 0>

When the members of G_1 are applied to the C(36,2) = 630 pairs of 4-tuples of R_1 the set of children produced is the universal relation $(E(3))^4$. This can be seen by noting that each of the nine possible pairs of values occurs in the first two position of the elements of R_1 and also in the last two positions of the elements of

Proceedings of the 25th International Symposium on Multiple-Valued Logic (ISMVL '95)
0730-3157/95 $10.00 © 1995 IEEE

R_1. Thus the pair $[g_3]$ and $[g_4]$, operating on R_1, produce the universal relation. If G_2 or G_3 were applied to R_0, the universal relation would be produced in the first generation.

The implication of this theorem is that the distinction among the operator sets G_1, G_2, and G_3 is small. If one begins with an initial relation of h-tuples and applies any one of the operator sets, one soon reaches a maximal closed relation. If one applies G_3 one arrives sooner that if one applies G_1, but in the latter case it requires at most $1 + [1 + \log_2 h]$ generations (where [n] denotes the greatest integer in n), as can be seen from the example above.

C. Closure Theorems

The discussion of the previous section leads to the natural question: if one begins with an initial relation R, and applies some set of reproduction operators, what is the maximal relation generated? Consider first a single pair of elements of a relation.

Let <s> and <t> be a pair of h-tuples. This pair (<s>,<t>) partitions the index set I = {1, 2, ..., h} into two subsets S and T, $S \cap T$ is empty, as follows:

$S = \{i | \text{ i in I}, s_i = t_i\}$ and $T = \{i | \text{ i in I}, s_i \neq t_i\}$.

Such a partition of I defines a relation C_S of degree h over E(k) which consists of all the h-tuples with the property that each pair of h-tuples agrees at exactly those indices which are the elements of S and disagrees at exactly those indices which are the elements of T. For example, suppose that over E(3), h = 9. Let

<s> = <0 1 2 0 1 2 0 1 2> and

<t> = <2 1 0 2 1 0 2 1 0>.

Hence S = {2, 5, 8} and T = {1, 3, 4, 6, 7, 9}. C_S consists of those 9-tuples which have a 1 in the second, fifth and eighth positions, that is, C_S contains all 9-tuples of the form <* 1 * * 1 * * 1 *>, where * is in {0, 1, 2}. Goldberg calls such a string a "schema" [2, pp. 18-19].

We say that <s> and <t> **coincide over S**, and that C_S is the **coincidence relation** generated by <s> and <t>. Each of the h-tuples <s> and <t> belongs to the coincidence relation that the pair generates. It might happen that a pair coincide on the empty set $\emptyset$, as when

<s> = <0 0 0 0 0 0 0 0 0> and

<t> = <1 1 1 1 1 1 1 1 1>.

In that case we define C_S to be the universal relation, $(E(k))^h$. If <s> = <t>, then S = I and C_S = {<s>}.

__Theorem 2__: If C_S is a coincidence relation over E(k) and if F is any set of reproduction operators, then F preserves C_S.

Proof: If S is not empty then by the definition of C_S there exist an <s> and <t> which generate C_S and a set S upon which <s> and <t> coincide. If <a> and <b> are any two elements of C_S then for all i in S, $a_i = b_i = s_i = t_i$. If $[f_j] = [f_{j1}, f_{j2}, ..., f_{jh}]$ in F is a reproduction operator then $f_{ji}(a_i,b_i) = a_i = b_i = s_i = t_i$, for all i in S. So each $[f_j]$ preserves C_S and F, which is the union of the $[f_j]$, preserves C_S. If S is empty then C_S is the universal relation and any operator preserves the universal relation. ∎

If R is a relation of degree h over E(k), then there are $|R|(|R| - 1)/2$ distinct pairs of members of R. Each pair induces a partition of the index set and has an incidence relation C_S.

__Theorem 3__: Let R be a relation of degree h over E(k). For each i, $1 \leq i \leq h$, let R(i) denote the set of elements in the i^{th} column of R. G_3 operating on R generates the relation R(1) x R(2) x ... x R(h).

Proof: Let <r_1>, <r_2>,...,<r_n> denote the elements of R. Let R1 = {[q]<r1>,r2> | [q] in G_3}. For 2 < j < n-1, let Rj = {[q]<r_{j+1}><r> | [q] in G_3, r in R_{j-1}}. Each set $R_1(i)$ contains those elements of E(k) which occur in either the i^{th} position of r_1 or the i^{th} position of r_2. Each set $R_j(i)$ contains those elements of E(k) which occur in either the i^{th} position of some <r_j>, $1 \leq j \leq$ n-1. ∎

__Corollary:__ If for each i, $1 \leq i \leq h$, R(i) = E(k) then G_3 generates the universal relation.

IV. Mutation

A. Definition

Informally, the idea of the mutation operation is that a h-tuple is changed by mutation in only one of its positions. Thus, the 9-tuple <0 1 2 0 1 2 0 1 2> might be changed by mutation into <0 1 2 0 1 1 0 1 2>.

A **mutation operator** $[m] = [m_1, m_2, ..., m_h]$ on $E^h(k)$ is a sequence of h unary functions on E(k) such that each m_i is the identity function except at most one, say m_j. The identity operator, denoted by [I], is a mutation operator. It is not in general true that if for all <x> in some space S of degree h, we have [m]<x> = <x>, then [m] = [I]. If, as is common, the distance D between two h-tuples in a search space S is taken to be Hamming distance, then for any mutation operator [m] and any h-tuple <x>, D([m]<x>, <x>) ≤ 1. We denote the set of mutation operators by M.

B. Completeness Theorem

In Theorem 3 and its Corollary we showed that any of the sets of crossover operators, G_1, G_2, G_3, operating on a relation R, generates a relation which is the Cartesian product of the sets that form the columns of R. If an element is not in the i^{th} column of the original relation it can never be generated by one of the crossover operators. It is the role of mutation to provide completeness.

__Theorem 4__: Let R be a relation of degree h over E(k). The set G_3 + M operating on R generates the universal relation.

Proof: In the proof of Theorem 3, note that M

Proceedings of the 25th International Symposium on Multiple-Valued Logic (ISMVL '95)
0730-3157/95 $10.00 © 1995 IEEE

operating on $R_1(i)$ produces $E(k)$. Thereafter each $R_j(i) = E(k)$. ■

<u>Corollary:</u> The set $G_1 + M$ operating on R generates the universal relation. The set $G_2 + M$ operating on R generates the universal relation.

V. Selection

A. Two Variants

Whilst reproduction and mutation are closely related, selection is a largely independent concept. In the form originally proposed by Holland [1], an initial population R_0 of size 2n is initially chosen at random in some search space S. The space S possesses some fitness function f, which is ordinarily real valued and which can, without loss of generality, be taken to have the range [0,1]. The elements of R_0 form n pairs which produce a set R_1 of 2n offspring by simple crossover. This set R_1 is the set of parents for the second generation.

Some later researchers objected to throwing away very fit individuals and so chose the 2n fittest individuals from $R_0 + R_1$. Often in this paradigm the 2n parents in R_0 produce a set of n children, each couple producing a single off-spring rather than twins. It is this last form of reproduction and selection that Cabrasawan and Wesselkamper used in [3].

B. Convergence Behavior

This second form of selection has a very troublesome property: for any reasonable fitness function the selection process converges to a fit population even if reproduction is replaced by the random selection of the off-spring at each generation. We now examine this behavior in a formal way.

Suppose that X is some search space endowed with a fitness function f: X -> [0,1]. Suppose that N is a large natural number. Consider the following algorithm to construct a set A_n of points in X.

<u>Selection Algorithm:</u> Let A_0 be a set of N randomly selected elements of X. For i = 0, 1, 2, ..., let B_i be a set of N randomly selected elements of X. Let + denote set theoretic union. Define:

$C_i = A_i + B_i$, (i = 0, 1, 2, ...);
$D_i = f(C_i)$, (i = 0, 1, 2, ...);
$E_i = \{x \mid x \text{ in } D_i, |E_i| = N, \text{for all } y \text{ in } D_i - E_i, x \geq y\}$;
$A_{i+1} = \{f^{-1}(x) \mid x \text{ in } E_i, f^{-1}(x) \text{ in } C_i\}$. ■

In the terminology of genetic algorithms suppose that we choose a population A_0 of N parents. Instead of generating their children by some crossover operation, choose the set B_0 of N children randomly. Using f, evaluate the fitness of $C_0 = A_0 + B_0$, calling the set of fitness values $D_0 = f(C_0)$. From D_0 choose E_0, the set consisting of the N largest values. Choose the new parent set A_1 to be those members of C_0 whose fitness values are in E_0. Continue this process, each time choosing B_i, the set of N children, randomly in X. The sets A_i form a sequence of sets whose mean fitness converges to 1. This convergence and the rate of convergence is ensured solely by the Selection Algorithm and conditions placed on the fitness function.

<u>**Theorem 5:**</u> If $x \neq y$ implies that $f(x) \neq f(y)$ and if the values f(x) are uniformly distributed over [0, 1], then the Selection Algorithm generates a sequence of sets A_i the lower bound of whose fitness converges to 1.

Proof: $f(A_0)$ and $f(B_0)$ each has lower bound 0 and mean 1/2. Likewise $D_0 = f(C_0)$ has 2N elements, lower bound 0, and mean 1/2. E_0 is the largest N points in D_0 with lower bound 1/2, and mean 3/4. A_1 consists of those points in C_0 whose images are in E_0. $D_1 = f(C_1)$ has 2N points, N/2 uniformly distributed in [0, 1/2] and 3N/2 uniformly distributed in [1/2, 1]. E_1 has lower bound 2/3 and mean 5/6. A_2 consists of those points in C_1 whose images are in E_1. Continuing in this way we immediately have the values:

Set	Lower Bound	Mean
$f(A_0)$	0/1	1/2
$f(A_1)$	1/2	3/4
$f(A_2)$	2/3	5/6
$f(A_3)$	3/4	7/8
...	...	...
$f(A_n)$	n/n+1	(2n+1)/(2n+2)

Thus, as n increases, both the lower bound and the mean of $f(A_i)$ converge to 1. ■

Similar theorems can be proved for other distributions of f in [0, 1]. The only thing that is really necessary for such a convergence theorem is that the values f(x) are dense in [0,1].

VI. Applications

The considerations in the previous sections of this paper allow us to explain the phenomena which we described in the first section. We reported that changes in the rate at which mutation occurred, from 0.0 through 0.1 had no effect whatever on the convergence of the mean fitness value of the population. This is explained by the result of Theorem 5. The convergence of the population to a population of very fit individuals was caused by the selection method that was employed. The convergence behavior is exactly the same if generalized crossover and mutation are replaced by a random selection of a set of N children at each step of the algorithm. It might happen in some situation that the employment of crossover and mutation improve the rate of convergence, but the fact of convergence is ensured by the selection algorithm used.

We reported that the convergence of the mean

Proceedings of the 25th International Symposium on Multiple-Valued Logic (ISMVL '95)
0730-3157/95 $10.00 © 1995 IEEE

fitness value of the population was logarithmic. It remained logarithmic when production of off-spring by crossover was replaced by random selection of off-spring. If the distribution of fitness values were uniform, we would have expected the sort of convergence rate described in the proof of Theorem 6. It appears that the logarithmic rate of convergence is caused by the fact that highly fit functions are significantly more common in the search space of 2-place functions which are Sheffer with constants, than unfit functions.

In addition, we note in Theorem 2 that reports that the three sets of crossover operators, G_1, G_2, and G_3, have significantly different power as generators of off-spring are greatly exaggerated. All three generate the same maximal relation of off-spring (cf. Theorem 3), albeit that G_3 generates that relation more quickly than G_2 and G_2 generates it more quickly than G_1.

References

[1] John H. Holland, **Adaptation in Natural and Artificial Systems,** Cambridge: The MIT Press, 1992

[2] David E. Goldberg, **Genetic Algorithms in Search, Optimization, and Machine Learning,** Reading: Addison- Wesley, 1989.

[3] Feb J. Cabrasawan and T. C. Wesselkamper, "Searching for Complete Functions over E(3) with Small Radii", **Proceeding of the 1994 International Symposium on Multiple-valued Logic**, Boston, May, 1994, pp. 172-6.

[4] Feb J. Cabrasawan, Joanna Lesniak, and T. C. Wesselkamper, "Genetic Algorithm Techniques for 3-valued Transistor Design", **Proceedings of the First IEEE Conference on Evolutionary Computation**, 1994, Vol. II, pp. 656-660.

[5] Masayuki Yamamura, Hiroshi Satoh, Shigenobu Kobayashi, "An Analysis of Crossover's Effect in Genetic Algorithms", **Proceedings of the First IEEE Conference on Evolutionary Computation**, Vol. II, pp. 613-618

[6] Zbigniew Michalwicz, **Genetic Algorithms + Data Structures = Evolution Programs**, New York: Springer-Verlag, 1994.

Proceedings of the 25th International Symposium on Multiple-Valued Logic (ISMVL '95)
0730-3157/95 $10.00 © 1995 IEEE