Phenotypes, Genotypes, and Operators in Evolutionary Computation

David B. Fogel
Natural Selection, Inc.
1591 Calle De Cinco
La Jolla, CA 92037
fogel@sunshine.ucsd.edu

ABSTRACT

Evolutionary computation can be conducted at various levels of abstraction (e.g., genes, individuals, species). Recent claims have been made that simulated evolution can be made more biologically accurate by applying specific genetic operators that mimic low-level transformations to DNA. This paper argues instead that the appropriateness of particular variation operators depends on the level of abstraction of the simulation. Further, including specific random variation operators simply because they have a similar form as genetic operators that occur in nature does not, in general, lead to greater fidelity in simulation.

1. Introduction

Evolution is characterized by distinct levels of hierarchy (e.g., species, individuals, chromosomes, genes), and quite naturally so is evolutionary computation. Typically, the elements in a simulated evolving population are considered to be analogous to *species* in evolutionary programming [9], *individuals* in evolution strategies [20], and *chromosomes* and *genes* in genetic algorithms [7, p. 2]. Recently, Collins [5] argued that evolutionary computation can be made more "biologically accurate" by including specific operators which mimic low-level changes that occur to DNA. Such a broad assertion ignores the level of abstraction of the simulation. The appropriateness of particular operators in a simulation derives from the level of abstraction and not necessarily from any specific resemblance to a mechanism found in natural evolution.

In general, models of natural phenomena can be constructed in two ways: bottom-up or top-down [10, pp. 255-258]. A *bottom-up* simulation emphasizes segregation of individual components and their local interactions, reducing the total system into successively smaller subsystems that are then analyzed in a piecemeal fashion. In contrast, *top-down* analyses emphasize extrinsically imposed forces and their associated physics as they pertain to the modeled system in its entirety. The appropriateness of different operators in an evolutionary computation follows from the adopted perspective, either bottom-up, emphasizing specific genetic mechanisms, or top-down, emphasizing phenotypic (behavioral) relationships. A careful examination of the nature of genotypes and phenotypes, and the mappings between them, reveals that the simple inclusion of genetic mechanisms as they occur in nature does not by consequence lead to more biologically accurate simulation. Indeed, the result may be opposite: the explanatory power of such a model of evolution may decrease, not increase, with the inclusion of successively more complicated genetic effects.

2. The Genotype, The Phenotype, and Lewontin's Mappings

Living organisms can be viewed as a duality of their genotype (the underlying genetic coding) and their phenotype (the manner of response contained in the behavior, physiology, and morphology of the organism). Lewontin [15] illustrated this distinction by specifying two state spaces: a populational genotypic (informational) space **G** and a populational phenotypic (behavioral) space **P**. Four functions map elements in **G** and **P** to each other (Figure 1). Atmar [1] modified these functions to be:

$$f_1: \mathbf{I} \times \mathbf{G} \rightarrow \mathbf{P}$$
$$f_2: \mathbf{P} \rightarrow \mathbf{P}$$
$$f_3: \mathbf{P} \rightarrow \mathbf{G}$$
$$f_4: \mathbf{G} \rightarrow \mathbf{G}.$$

The function f_1, *epigenesis*, maps the element $g_1 \in \mathbf{G}$ into the phenotypic space **P** as a particular collection of phenotypes p_1 whose development is modified by its environment, an indexed set of symbols $(i_1, ..., i_k) \in \mathbf{I}$, where **I** is the set of all such environmental sequences. The function f_2, *selection*, maps phenotypes p_1 into p_2. As natural selection operates only on the phenotypic expressions of the genotype [17, p. 131; 12, p. 431], the underlying coding g_1 is not involved in the function f_2. The function f_3, *genotypic survival*, describes the effects of selection and migration processes on **G**. Function f_4,

mutation, maps the representative codings $g_2 \in \mathbf{G}$ to the point $g'_1 \in \mathbf{G}$. This function represents the "rules" of mutation and recombination, and encompasses all genetic changes. With the creation of the new population of genotypes g'_1, one generation is complete. Evolutionary adaptation occurs over successive iterations of these mappings.

The mapping between the genotype and the phenotype is characterized by pleiotropy and polygeny (Figure 2). *Pleiotropy* is the effect that a single gene may simultaneously affect several phenotypic traits. *Polygeny* is the effect that a single phenotypic characteristic of an individual may be determined by the simultaneous interaction of many genes. Pleiotropy and polygeny prohibit any useful reductionist simplification of the genotype-phenotype mapping; the mapping is not linearly separable. Single genetic changes can result in multiple phenotypic changes and individual phenotypic traits cannot be assigned to a single genetic cause (with the exception of gene defects, see [2]).

3. Levels of Hierarchy in Evolutionary Computation

All efforts in evolutionary computation involve population-based search with random variation and selection. But the multiple levels of hierarchy in evolution and the dual nature of individuals in terms of their genotype and phenotype suggest various methods for simulating evolution. In particular, attention may be focused on the genes of individuals, on the behavior of individuals, or on the behavior of reproducing populations (species). Each of these perspectives are represented by the efforts in genetic algorithms, evolution strategies, and evolutionary programming, respectively.

Within the framework of genetic algorithms, attention is given to the genotypes of individuals, and the elements (data structures) in an evolving population are considered analogous to chromosomes and genes [7, p. 2]. Each structure, which in this case is purely genotypic, is decoded into a phenotypic expression which is in turn evaluated in light of a fitness function. Selection acts to replicate the structures in the population in proportion to their relative fitness. New structures are created by applying operators that mimic the forms of transformations that occur to natural chromosomes and genes (e.g., crossover, inversion, mutation).

Within evolution strategies, attention is instead given to the phenotypes of individuals, and the elements in an evolving population are abstracted as vectors of behavioral traits [20]. Each vector of traits is evaluated in light of a fitness function. Selection acts to eliminate those behaviors having a performance that is below a prescribed threshold. New vectors are created by applying operators which mimic the functional change in pheno-

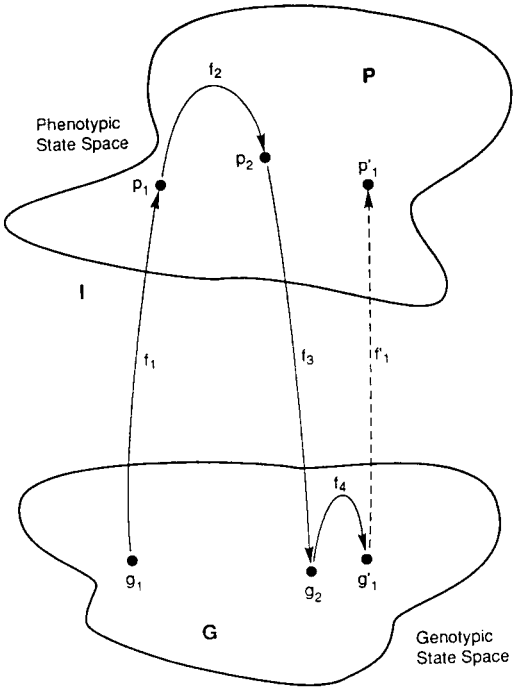

Figure 1. The evolution of a population within a single generation. Evolution can be viewed as occurring as a succession of four mapping functions (*epigenesis*, *selection*, *genotypic survival*, and *mutation*) relating the genoptypic information state space and the phenotypic behavioral state space.

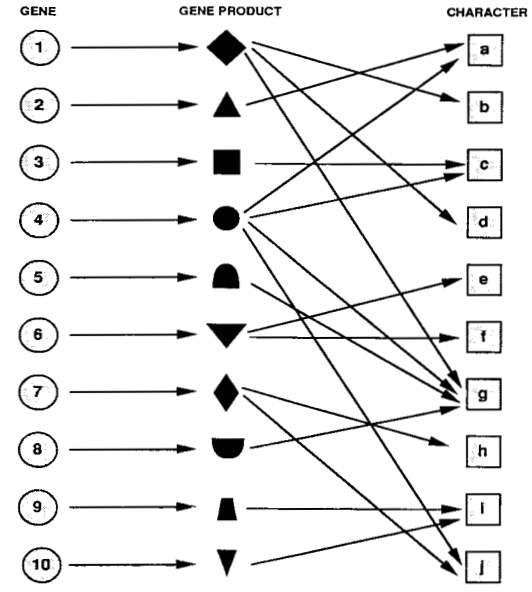

Figure 2. The effects of *pleiotropy* (single gene—multiple phenotypic expressions) and *polygeny* (single phenotypic character—multiple genes) (after [16, p. 265]).

typic traits in individuals. For example, for continuous-valued traits, the behavioral change may follow a zero-mean Gaussian random variable. Further, the variability of the change may be determined in part by genetic structures (i.e., *strategy parameters*) which accompany each vector of phenotypic traits. These genetic structures may undergo varieties of recombination and mutation operators.

Within evolutionary programming, attention is given to the phenotypes of entire reproductive populations (species), and the elements in an evolving population can again be abstracted as vectors of behavioral traits [10]. In a similar manner as with individual traits, each vector of behavioral traits is evaluated in light of a fitness function and selection acts to eliminate those behaviors having a performance that is below a prescribed threshold. New vectors are created by applying operators that mimic the functional change in phenotypic traits in populations. Again, the variability of the change may be determined in part by genetic strategy parameters. But these genetic structures are only subject to individual random variation and do not undergo varieties of recombination operators because there is no recombination between species (using the *biological species concept* of Mayr [18, p. 318].

4. Phenotypic and Genotypic Operators

The level of abstraction in a simulation of evolution dictates the appropriateness of phenotypic or genotypic operators. In genetic-based simulations (e.g., genetic algorithms, [7, 14]), operators which mimic the form of genetic transformation in natural evolution are applied to abstracted genotypes. In phenotypic-based simulations (e.g., evolution strategies or evolutionary programming, [20, 10]), operators are applied to abstracted phenotypes (primarily, but also to genotypes) and are chosen so as to generate a particular distribution of new behaviors (see below). Figure 3 indicates the functional flow of these procedures in terms of Lewontin's mappings (Figure 1).

The reasonableness of any particular operator, in terms of its biological fidelity, can be assessed by placing the operator in the context of the level of abstraction of the simulation. For example, in general, it would not be appropriate to apply recombination operators to two parental phenotypes. Although there are some exceptions (e.g., [4, pp. 271-272]), the collection of phenotypic traits of most individuals is not determined simply as a random discrete mixing or (weighted) average of the phenotypic traits of their parents. Therefore, for a real-valued optimization problem, if the representation chosen is a vector of real values which are inputs to the fitness function, then these values are phenotypic, and the crossing over of these real values is of dubious merit in terms of the degree to which the simulation abstracts the natural process of evolution.

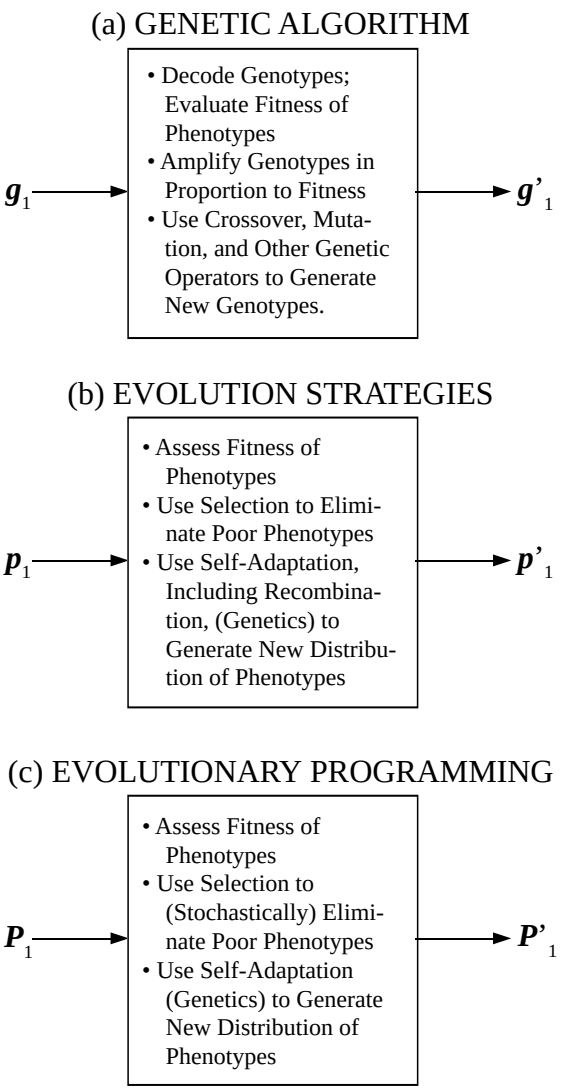

Figure 3. The functional flow of (a) genetic algorithms, (b) evolution strategies, and (c) evolutionary programming in terms of the elements in Lewontin's Mappings (Figure 1).Genetic algorithms operate from genotype to genotype; evolution strategies and evolutionary programming operate from phenotype to phenotype. The symbols $\boldsymbol{p}_1$ and $\boldsymbol{P}_1$ are used to emphasize evolution strategies and evolutionary programming modeling phenotypes of individuals or reproducing populations (species), respectively. Recombination may be applied to self-adaptative parameters in evolution strategies, but is not applicable in evolutionary programming. In genetic algorithms, new genotypes are generated by emulating specific genetic mechanisms as observed in nature. In evolution strategies and evolutionary programming, new phenotypes are generated so as to take on a desired distribution. The phenotypes may not be explicit, but may instead be a function of genetic structures (see Figures 5 and 6).

Genetic algorithms apply variation solely to genetic structures, not phenotypic structures (Goldberg [11, p. 7] offered that genetic algorithms “work with a coding

of the parameter set, not the parameters themselves"). In contrast, evolution strategies, and evolutionary programming may be purely phenotypic (operating only on parameters that are placed directly into the fitness function and evaluated), or purely genetic (operating on parameters that are used to generate behavior in light of external "environmental" factors), or both. The chosen level of hierarchy should be used to determine an appropriate set of operators for manipulating the data structures that encode solutions in evolutionary computation.

For example, consider the problem of optimizing the weights of a fixed-topology multilayer neural network in a pattern classification task (Figure 4). The weights, although they may assume real values, are not phenotypic; they are purely genetic. The weights are not the behavior of the neural network. Only when the available exemplars are provided and the computational nodes and connections of the network are specified can the weights be used to determine the behavior of the network. An error function is used to assess the network's behavior, and this assessment cannot be apportioned back to individual weights; only the entire set of weights can be evaluated in terms of the collective set of behaviors they generate in light of the set of exemplars. Thus any operator that varies the weights must be a genetic, not a phenotypic, operator.

Within evolution strategies and evolutionary programming, the general approach to neural network training is to vary the weights by a multivariate Gaussian random variable (e.g., [19]), just as is the normal procedure for simple real-valued function optimization problems. But the rationale for selecting the Gaussian operator in the case of optimizing neural weights is much different. The focus of attention is on the changes that are generated to the network's output (i.e., its behavior) as a function of the changes to weights in a neural network (i.e., its genetics). Gaussian random variables are used because they can generate nearly Gaussian changes in the network's output (Figure 5).

Similar choices of operators can be made for other problems with discrete representations. For example, in the traveling salesman problem, Fogel [8] represented candidate tours by a simple ordering of the cities to be visited. The listing itself was purely genetic; it had no behavior. Only when the listing was interpreted as an ordering of cities to be visited, with an implicit return to the city of origin, did the listing imply a circuit (i.e., the behavior), which was then evaluated in light of its total Euclidean distance.

Fogel [8] used an inversion operator to generate new solutions by reversing the ordering of subsections of a parent tour. By varying the length of the inversion, a range of possible new tour lengths could be generated, and the variance of the distribution of new tour lengths could be controlled by the size of the inversion (Figure 6). Note that inversion was not chosen in [8] because it occurs in nature, nor is there any reason to believe that the function of any natural genetic inversion was captured by the implementation of this inversion operator on the traveling salesman problem.

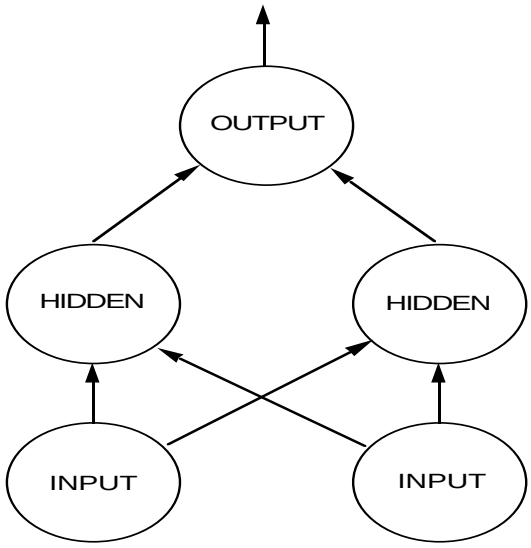

Figure 4. A simple feedforward neural network with two input nodes, two hidden nodes, and a single output node. Each hidden node performs a dot product of the input values and the weights on each associated connection, subtracts off a bias term (not shown) and then passes the result through a sigmoid filter, $f(x) = 1/(1+e^{-x})$. The output node performs the same dot product and bias operation but does not pass the result through a sigmoid filter. This architecture was used to generate the data displayed in Figure 5.

5. Discussion

Natural selection is the predominant mediating evolutionary force that prevails in shaping the phenotypic characters of individuals and species in the vast majority of situations encountered in nature [3, 13, 18, 21, p. 75; and others]. Selection, as illustrated by Lewontin [15] (Figure 1), acts only on the phenotypes of individuals and populations, which are determined by a complex and nonlinear function (including pleiotropy and polygeny) of the interactions between their genotypes and the environment.

It is therefore not reasonable to expect that the function (i.e., phenotypic effect) of a given genetic operator as it occurs in nature can be easily replicated in a simulation simply by including an idealized version of the operator. In a simulation, both the genetic information at hand and the environment with which the genotype interacts to generate a phenotype ($\mathbf{I} \times \mathbf{G} \rightarrow \mathbf{P}$) are qualitatively different (e.g., orders of magnitude less complex) from their natural analogues. Any belief that the simple inclusion of another genetic operator with a bio-

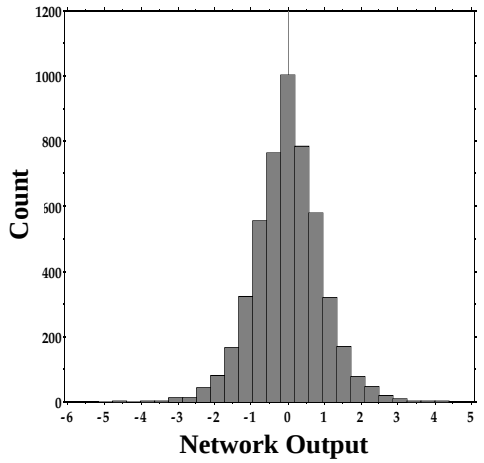

Figure 5. A histogram from 5000 samples of the output from the neural network depicted in Figure 4 when varying the weights and bias terms by a multivariate Gaussian random variable. The initial weights of the network were determined by sampling from a uniform random variable over the range [-0.5,0.5]. The inputs were set equal to 1.0. The initialized output of the network was 0.0275. Zero mean Gaussian noise was imposed on each weight and bias, with a variance of 0.5. The resulting behavior (phenotype) of the network was approximately Gaussian. The variance of the phenotype could be adjusted by varying the size of the mutations.

logical analogy generates a more accurate simulation is delusory. In fact, the functional effect of the operator may be entirely different in the simulation, and selection may then act to drive a modelled population in a trajectory that has no correspondence to any natural observations.

This reveals an inherent limitation of genetic algorithms. As bottom-up reductionist models of evolution they are not broadly useful for making predictions about natural evolution. The behavior of any evolutionary computation cannot be anticipated by analyzing its components (e.g., a specific genetic operator or representation) in isolation; the interactions of the components are far more important and these are invisible at the level of individual components. The inclusion of any new operator or any other change in the parameters of the simulation may lead to unpredictable changes in the simulation's behavior (i.e., *emergent properties*). By overemphasizing low-level operators and components, the fundamental effect of selection on behaviors in evolutionary processes is lost.

Extreme care must be used in extrapolating from any model of natural evolution. But top-down models that emphasize individual and species behavior (e.g., evolution strategies and evolutionary programming), and the adaptation and diversity of that behavior in light of selection, offer the possibility for making reasonable pre-

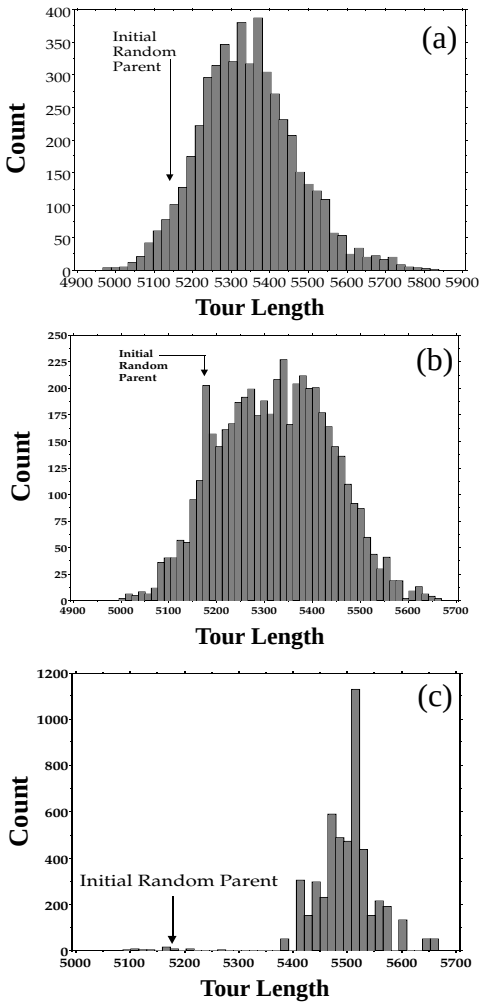

Figure 6. Histograms of 5000 samples of the tour length generated by inverting a subsection of a random tour of 100 cities. The initial tour was generated by selecting (*x*,*y*) coordinates of 100 cities with *x* and *y* distributed uniformly over the integers [1,100]. The order in which the cities were generated was taken as the initial circuit and had a length of 5171.65 units. An inversion (mutation) was generated by (1) randomly choosing a city in the list, (2) choosing another city that was a chosen number of steps away from the first city, and (3) reversing the order of the list between the two chosen cities. Three different length inversions were applied in separate experiments. The histograms depict the frequency of the resulting tour lengths when applying an inversion of (a) 40 units, (b) 50 units, and (c) 2 units (adjacent cities). The larger mutations (a,b) generated a wider range of tour lengths, and increased (empirically determined) probabilities of improving the randomly generated initial tour (i.e., 0.076, 0.095, and 0.004 for (a), (b), (c), respectively). Further, the larger mutations had the potential to generate shorter tours. The results indicate that mutations can be chosen in light of the behavioral effect they portend. The results also suggest that the appropriate mutation "step size" can be determined empirically at different stages of simulated evolution.

dictions of the patterns of phenotypes in response to selective pressures, regardless of the genetic mechanisms involved in the natural process. It may be worthwhile to recall that Darwin [6] offered his theories of evolution in the complete absense of any knowledge of genetics.

6. Acknowledgments

The author would like to thank P.J. Angeline and L.J. Fogel for their comments and criticisms of this paper.

7. References

[1] W. Atmar, "On the Rules and Nature of Simulated Evolutionary Programming," in *Proceedings of the First Annual Conference on Evolutionary Programming*, eds. D.B. Fogel and W. Atmar, Evol. Prog. Society, La Jolla, CA, 1992, pp. 17-26.

[2] W. Atmar, "Notes on the Simulation of Evolution," *IEEE Trans. Neural Networks*, Vol. 5:1, 1994, pp. 130-147.

[3] C.F. Brunk, "Darwin in an Age of Molecular Revolution," *Contention*, Vol 1:1, 1991, pp. 131-150.

[4] N.A. Campbell, *Biology*, 2nd ed., Bejamin/ Cummings Pub. Co., Inc., Redwood City, CA, 1990.

[5] R.J. Collins, "Artificial Evolution and the Paradox of Sex," in *Computing with Biological Metaphors*, ed. R. Paton, Chapmann & Hall, London, 1994, pp. 244-263.

[6] C. Darwin, *On the Origin of Species by Means of Natural Selection or the Preservations of Favored Races in the Struggle for Life*, Murray, London, 1859.

[7] L. Davis (ed.), *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, NY, 1991.

[8] D.B. Fogel, "Empirical Estimation of the Computation Required to Discover Approximate Solutions to the Traveling Salesman Problem Using Evolutionary Programming," in *Proceedings of the Second Annual Conference on Evolutionary Programming*, eds. D.B. Fogel and W. Atmar, Evol. Prog. Society, La Jolla, CA, 1993, pp. 56-61.

[9] D.B. Fogel, "An Introduction to Simulated Evolutionary Optimization," *IEEE Trans. Neural Networks*, Vol. 5:1, 1994, pp. 3-14.

[10] D.B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*, IEEE Press, Piscataway, NJ, 1995.

[11] D.E. Goldberg, *Genetic Algorithms in Search Optimization and Machine Learning*, Addison-Wesley, Reading, MA, 1989.

[12] D.L. Hartl and A.G. Clark, *Principles of Population Genetics*, 2nd ed., Sinauer, Sunderland, MA, 1989.

[13] A. Hoffman, *Arguments on Evolution: A Paleontologist's Perspective*, Oxford, NY, 1989.

[14] J.H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, 1975.

[15] R.C. Lewontin, *The Genetic Basis of Evolutionary Change*, Columbia University Press, NY, 1974.

[16] E. Mayr, *Animal Species and Evolution*, Belknap Press, Cambridge, MA, 1963.

[17] E. Mayr, *Populations, Species, and Evolution*, Belknap Press, Cambridge, MA, 1970.

[18] E. Mayr, *Toward a New Philosophy of Biology: Observations of an Evolutionist*, Belknap Press, Cambridge, MA, 1988.

[19] V.W. Porto, D.B. Fogel, L.J. Fogel, "Alternative Neural Network Training Methods," *IEEE Expert*, Vol. 10;3, June, 1995, pp. 16-22.

[20] H.-P. Schwefel, *Evolution and Optimum Seeking*, John Wiley, NY, 1995.

[21] E.O. Wilson, *The Diversity of Life*, W.W. Norton, NY, 1992.